

09/673503  
日本国特許庁  
PATENT OFFICE  
JAPANESE GOVERNMENT

PCT/JP99/02049

19.04.99

別紙添付の書類に記載されている事項は下記の出願書類に記載されて  
いる事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed  
with this Office.

出願年月日  
Date of Application:

1998年10月27日

REC'D 14 JUN 1999  
WIPO PCT

出願番号  
Application Number:

平成10年特許願第305336号

出願人  
Applicant(s):

大見 忠弘  
株式会社ウルトラクリーンテクノロジー開発研究所

E.O

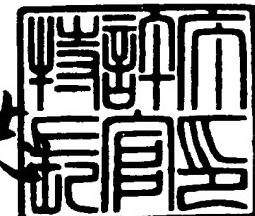
**PRIORITY  
DOCUMENT**

SUBMITTED OR TRANSMITTED IN  
COMPLIANCE WITH RULE 17.1(a) OR (b)

1999年 5月28日

特許庁長官  
Commissioner,  
Patent Office

伴佐山達志



出証番号 出証特平11-3033370

【書類名】 特許願  
【整理番号】 XY10517  
【提出日】 平成10年10月27日  
【あて先】 特許庁長官殿  
【国際特許分類】 H01L 21/00  
【発明の名称】 データ圧縮装置および方法、データ伸長装置および方法  
、データ圧縮伸長システムおよび方法、コードブックの  
作成方法、記録媒体  
【請求項の数】 72  
【発明者】  
【住所又は居所】 宮城県仙台市青葉区米ヶ袋2の1の17の301  
【氏名】 大見 忠弘  
【発明者】  
【住所又は居所】 千葉県千葉市美浜区稻毛海岸5丁目5-2-206  
【氏名】 小谷 光司  
【発明者】  
【住所又は居所】 東京都足立区加平二丁目12番5号  
【氏名】 中田 明良  
【発明者】  
【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大学内  
【氏名】 今井 誠  
【発明者】  
【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大学内  
【氏名】 韶田 正宏  
【発明者】  
【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大学内  
【氏名】 森本 達郎  
【発明者】  
【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大学内

【氏名】 米澤 岳美

## 【発明者】

【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大學内

【氏名】 野沢 俊之

## 【発明者】

【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大學内

【氏名】 中山 貴裕

## 【発明者】

【住所又は居所】 宮城県仙台市青葉区荒巻字青葉（無番地）東北大學内

【氏名】 藤林 正典

## 【発明者】

【住所又は居所】 東京都文京区本郷4丁目1番4号 株式会社ウルトラクリーンテクノロジー開発研究所内

【氏名】 新田 雄久

## 【特許出願人】

【識別番号】 000205041

【氏名又は名称】 大見 忠弘

## 【特許出願人】

【識別番号】 596089517

【氏名又は名称】 株式会社ウルトラクリーンテクノロジー開発研究所

## 【代理人】

【識別番号】 100090273

## 【弁理士】

【氏名又は名称】 國分 孝悦

【電話番号】 03-3590-8901

## 【先の出願に基づく優先権主張】

【出願番号】 平成10年特許願第124286号

【出願日】 平成10年 4月17日

【先の出願に基づく優先権主張】

【出願番号】 平成10年特許願第208364号

【出願日】 平成10年 7月23日

【手数料の表示】

【予納台帳番号】 035493

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9814197

【包括委任状番号】 9814281

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 データ圧縮装置および方法、データ伸長装置および方法、データ圧縮伸長システムおよび方法、コードブックの作成方法、記録媒体

【特許請求の範囲】

【請求項1】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

上記ベクトルを構成するブロックを、データ位置を1次元的に配列したラインブロックとすることを特徴とするデータ圧縮装置。

【請求項2】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

上記圧縮対象のベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するようにするブロックシフト手段を備えたことを特徴とするデータ圧縮装置。

【請求項3】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項2に記載のデータ圧縮装置。

【請求項4】 少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長装置において、

上記コードブックの中から探し出したコードベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして割り当てるブロックシフト手段を備えたことを特徴とするデータ伸長装置。

【請求項5】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項4に記載のデータ伸長

装置。

【請求項6】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、データ圧縮時には、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するとともに、データ伸長時には、上記コードブックの中から上記コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ圧縮伸長システムにおいて、

上記データ圧縮時において、上記圧縮対象のベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するようとする第1のブロックシフト手段と、

上記データ伸長時において、上記コードブックの中から探し出したコードベクトルのブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして割り当てる第2のブロックシフト手段との少なくとも何れかを備えたことを特徴とするデータ圧縮伸長システム。

【請求項7】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項6に記載のデータ圧縮伸長システム。

【請求項8】 上記第1のブロックシフト手段によるブロックシフトと、上記第2のブロックシフト手段によるブロックシフトとは、シフト量が同じで方向が互いに逆の関係にあることを特徴とする請求項6または7に記載のデータ圧縮伸長システム。

【請求項9】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を

各時刻毎に出力する第1のベクトル量子化手段と、

上記第1のベクトル量子化手段により各時刻毎に出力されたコード列を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とするデータ圧縮装置。

【請求項10】 上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、上記各時刻におけるデータ間で対応するアドレスのコードどうしをまとめて新たなベクトルを複数生成するコード列再配列手段を備えたことを特徴とする請求項9に記載のデータ圧縮装置。

【請求項11】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、

上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、ある時刻におけるデータのコード列を基準コード列とし、上記基準コード列と他の時刻におけるデータのコード列との間において対応するアドレスのコード間で差分をとった結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とするデータ圧縮装置。

【請求項12】 上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、ある時刻におけるデータのコード列を基準コード列とし、上記基準コード列と他の時刻におけるデータのコード列との間において対応するアドレスのコード間で差分をとり、上記対応するアドレスの差分どうしをまとめて

新たなベクトルを複数生成するコード列再配列手段を備えたことを特徴とする請求項11に記載のデータ圧縮装置。

【請求項13】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、

上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、時間軸方向に隣接するデータの間において対応するアドレスのコード間で差分をとった結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とするデータ圧縮装置。

【請求項14】 上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、時間軸方向に隣接するデータの間において対応するアドレスのコード間で差分をとり、上記対応するアドレスの差分どうしをまとめて新たなベクトルを複数生成するコード列再配列手段を備えたことを特徴とする請求項13に記載のデータ圧縮装置。

【請求項15】 少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長装置において、

請求項9または10に記載のデータ圧縮装置により生成されたコード列に対応するコードベクトルを上記第2のコードブックから夫々探し出して出力する第2のデコード手段と、

上記第2のデコード手段により出力された複数のコードベクトルを構成するコ

ード列を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、上記新たなベクトルを構成するコード列に対応するコードベクトルを上記第1のコードブックから夫々探し出して出力する第1のデコード手段とを備えたことを特徴とするデータ伸長装置。

【請求項16】 少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長装置において、

請求項11または12に記載のデータ圧縮装置により生成されたコード列に対応するコードベクトルを上記第2のコードブックから夫々探し出して出力する第2のデコード手段と、

上記第2のデコード手段により出力された複数のコードベクトルを構成するコード列と上記基準コード列との間で対応するアドレスのコードどうしを加算した結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、上記新たなベクトルを構成するコード列に対応するコードベクトルを上記第1のコードブックから夫々探し出して出力する第1のデコード手段とを備えたことを特徴とするデータ伸長装置。

【請求項17】 少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長装置において、

請求項13または14に記載のデータ圧縮装置により生成されたコード列に対応するコードベクトルを上記第2のコードブックから夫々探し出して出力する第2のデコード手段と、

上記第2のデコード手段により出力された複数のコードベクトルを構成するコード列について、コードをアドレス順に加算していった夫々の結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、上記新たなベクトルを構成するコード列に対応するコードベクトルを上記第1のコードブックから夫々探し出して出力する第1のデコード手段とを備えたことを特徴

とするデータ伸長装置。

【請求項18】 上記複数の新たなベクトルを生成する際に、上記新たなベクトル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようにする時間軸シフト手段を備えたことを特徴とする請求項9～14の何れか1項に記載のデータ圧縮装置。

【請求項19】 上記複数の新たなベクトルを生成する際に、上記新たなベクトル内の少なくとも近接する要素を時間軸方向にずらして配列するようにする時間軸シフト手段を備えたことを特徴とする請求項15～17の何れか1項に記載のデータ伸長装置。

【請求項20】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータ列のうち、各時刻におけるデータ間で対応するアドレスのデータどうしをまとめて複数のベクトルとし、それらのベクトルに類似したコードベクトルをコードブックから夫々探し出し、それらに対応するコード列を出力するベクトル量子化手段を備えたことを特徴とするデータ圧縮装置。

【請求項21】 上記複数のベクトルを取り出す際に、上記ベクトル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようにする時間軸シフト手段を備えたことを特徴とする請求項20に記載のデータ圧縮装置。

【請求項22】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

上記圧縮対象から少なくとも1つ以上の特徴量を抽出し、特徴量データと上記圧縮対象から上記特徴量を除いた基本パターンデータとに分離する分離手段を備え、

上記分離した特徴量データと基本パターンデータとに対してそれぞれ独立にベクトル量子化を行うようにしたことを特徴とするデータ圧縮装置。

【請求項23】 上記分離手段によって時間軸に沿って変化する圧縮対象から分離された各時刻における基本パターンデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを基本パターン用の第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、

上記第1のベクトル量子化手段により各時刻毎に出力されたコード列を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを基本パターン用の第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段と、

上記分離手段によって上記時間軸に沿って変化する圧縮対象から分離された各時刻における特徴量を構成するデータ列を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを特徴量用の第3のコードブックから夫々探し出し、それらに対応するコード列を出力する第3のベクトル量子化手段とを備えたことを特徴とする請求項22に記載のデータ圧縮装置。

【請求項24】 少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長装置において、

請求項22または23に記載のデータ圧縮装置より出力された基本パターンに関するコード列に基づいて元の基本パターンデータを再現する基本パターン用の第1のデコード手段と、

請求項22または23に記載のデータ圧縮装置より出力された特徴量に関するコード列に基づいて元の特徴量データを再現する特徴量用の第2のデコード手段と、

上記第1、第2のデコード手段の出力結果を合成して元データを再現する合成

手段とを備えたことを特徴とするデータ伸長装置。

【請求項25】少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルをコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力するベクトル量子化手段と、

上記ベクトル量子化手段により各時刻毎に出力されるコード列について、時間軸方向に隣接するデータの対応するアドレスどうしでコード間の相関を夫々とり、相関が所定値より小さいアドレスのコードのみを出力する出力制御手段とを備えたことを特徴とするデータ圧縮装置。

【請求項26】少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象のカラー画像より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、

輝度信号用のコードブックと色信号用のコードブックとを備え、上記色信号用のコードブックよりも上記輝度信号用のコードブックの方により多くのコードベクトルを割り当てるようにしたことを特徴とするデータ圧縮装置。

【請求項27】上記輝度信号であるY信号、および上記色信号であるU信号、V信号の圧縮レートが4:1:1もしくは4:2:2であることを特徴とする請求項26に記載のデータ圧縮装置。

【請求項28】少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化的処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、

上記初期のコードブックとして、とり得る値の最小値から最大値までデータ値が連続的に変化するパターンのコードブックを用いたことを特徴とするコードブックの作成方法。

【請求項29】少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化的処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、

上記1回の処理で更新する範囲を上記仮想的な2次元平面上で1次元的に適用し、かつ更新回数の増加と共に範囲を減少させることを特徴とするコードブックの作成方法。

【請求項30】少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化的処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、

上記更新係数の初期値を0.3~1の範囲内の何れかの値とし、かつ更新回数の増加と共に上記更新係数の値を減少させることを特徴とするコードブックの作成方法。

【請求項31】少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化的処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、

上記初期のコードブックとして、とり得る値の最小値から最大値までデータ値が連続的に変化するパターンのコードブックを用い、

上記1回の処理で更新する範囲を上記仮想的な2次元平面上で1次元的に適用し、かつ更新回数の増加と共に範囲を減少させ、

上記更新係数の初期値を0.3～1の範囲内の何れかの値とし、かつ更新回数の増加と共に上記更新係数の値を減少させるようにしたことを特徴とするコードブックの作成方法。

【請求項32】少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化の処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、

複数のサンプルデータに対して独立にコードブックを最適化し、得られた複数のコードブックを合成して新たなコードブックを作成するようにしたことを特徴とするコードブックの作成方法。

【請求項33】上記複数のサンプルデータに対して独立にコードブックを最適化する際に、請求項28～31の何れか1項に記載の方法によりコードブックを作成するようにしたことを特徴とする請求項32に記載のコードブックの作成方法。

【請求項34】コードブック内の各コードベクトルに関してあらかじめ求められた特徴量を記憶する特徴量記憶手段と、

上記圧縮対象より抽出されるベクトルの特徴量を求める特徴量演算手段と、

上記特徴量記憶手段に記憶されている各コードベクトルの特徴量と上記特徴量演算手段により求められた圧縮対象ベクトルの特徴量に基づいて、上記各コードベクトルのそれについて上記圧縮対象ベクトルとの類似度を求める演算を省略するかどうかを決定する演算省略手段とを備えたことを特徴とする請求項1～3、9～14、18、20～21、25～27の何れか1項に記載のデータ圧縮装置。

【請求項35】上記特徴量は、ベクトルを構成するデータ列の値の総和、平均値もしくは直流成分であることを特徴とする請求項34に記載のデータ圧縮

装置。

【請求項36】 上記特徴量は、ベクトルを構成するデータ列のうちの一部の要素の値を、とり得る値の中間値を基準として反転するように操作した後の値の総和、平均値もしくは直流成分であることを特徴とする請求項34に記載のデータ圧縮装置。

【請求項37】 上記特徴量は、ベクトルを構成するデータ列の値のブロック内での変化の方向であることを特徴とする請求項34に記載のデータ圧縮装置。

【請求項38】 上記特徴量は、ベクトルを構成するデータ列の値のブロック内での変化の態様であることを特徴とする請求項34に記載のデータ圧縮装置。

【請求項39】 上記演算省略手段は、あるコードベクトルの特徴量と上記圧縮対象ベクトルの特徴量との差分絶対値を求める差分絶対値演算手段と、

上記差分絶対値演算手段により求められた差分絶対値が、他のコードベクトルについて既に求められている類似度を表す値の最小値よりも大きい場合に、上記あるコードベクトルについての類似度の演算を省略する省略判定手段とを備えることを特徴とする請求項34～38の何れか1項に記載のデータ圧縮装置。

【請求項40】 コードブック内の各コードベクトルに関してあらかじめ求められた異なる種類の特徴量を記憶する特徴量記憶手段と、

上記圧縮対象より抽出されるベクトルについて異なる種類の特徴量を求める特徴量演算手段と、

上記特徴量記憶手段に記憶されている各コードベクトルに関する異なる種類の特徴量と、上記特徴量演算手段により求められた圧縮対象ベクトルに関する異なる種類の特徴量とに基づいて、上記各コードベクトルのそれぞれについて上記圧縮対象ベクトルとの類似度を求める演算を省略するかどうかを決定する演算省略手段とを備えたことを特徴とする請求項1～3、9～14、18、20～21、25～27の何れか1項に記載のデータ圧縮装置。

【請求項41】 少なくとも1つ以上のコードブックを保持しておくコードブックサーバと、データ圧縮システムと、データ伸長システムとを備えたデータ

圧縮伸長システムであって、

上記コードブックサーバは、上記データ圧縮システムおよびデータ伸長システムからの要求に従って、保持している何れかのコードブックを供給することを特徴とするデータ圧縮伸長システム。

【請求項4 2】 上記コードブックサーバは、少なくとも1つ以上のコードブックを保持するコードブック保持手段と、

上記データ圧縮システムおよびデータ伸長システムからの要求に合ったコードブックを生成するコードブック生成手段と、

上記データ圧縮システムおよびデータ伸長システムからの要求に応じて、上記コードブック保持手段に保持されているコードブックもしくは上記コードブック生成手段により生成されたコードブックを供給するコードブック管理手段とを備えることを特徴とする請求項4 1に記載のデータ圧縮伸長システム。

【請求項4 3】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

上記ベクトルを構成するブロックを、データ位置を1次元的に配列したラインブロックとしたことを特徴とするデータ圧縮方法。

【請求項4 4】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

上記圧縮対象のベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するようにしたことを特徴とするデータ圧縮方法。

【請求項4 5】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項4 4に記載のデータ圧縮方法。

【請求項4 6】 少なくとも1つ以上のデータを有するデータ列をベクトル

とし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ伸長方法において、

上記コードブックの中から探し出したコードベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして割り当てるようとしたことを特徴とするデータ伸長方法。

【請求項47】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項46に記載のデータ伸長方法。

【請求項48】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、データ圧縮時には、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するとともに、データ伸長時には、上記コードブックの中から上記コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てることによって元データを再現するデータ圧縮伸長方法において、

上記データ圧縮時において、上記圧縮対象のベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するとともに、

上記データ伸長時において、上記コードブックの中から探し出したコードベクトルのブロックを、少なくとも垂直方向に近接するブロック間では、上記データ圧縮時とシフト量が同じで方向が互いに逆の関係となるように空間的位置を水平方向にずらして割り当てるようとしたことを特徴とするデータ圧縮伸長方法。

【請求項49】 上記ベクトルを構成するブロックは、データ位置を1次元的に配列したラインブロックであることを特徴とする請求項48に記載のデータ圧縮伸長方法。

【請求項50】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応する

コードを出力するデータ圧縮方法において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化ステップと、

上記第1のベクトル量子化ステップで各時刻毎に出力されたコード列のうち、上記各時刻におけるデータ間で対応するアドレスのコードどうしをまとめて新たなベクトルを複数生成するコード列再配列ステップと、

上記コード列再配列ステップで生成された複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化ステップとを有することを特徴とするデータ圧縮方法。

【請求項51】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化ステップと、

上記第1のベクトル量子化ステップで各時刻毎に出力されたコード列のうち、ある時刻におけるデータのコード列を基準コード列とし、上記基準コード列と他の時刻におけるデータのコード列との間において対応するアドレスのコード間で差分をとり、上記対応するアドレスの差分どうしをまとめて新たなベクトルを複数生成するコード列再配列ステップと、

上記コード列再配列ステップで生成された複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化ステップとを有することを特徴とするデータ圧縮方法。

【請求項52】少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化ステップと、

上記第1のベクトル量子化ステップで各時刻毎に出力されたコード列のうち、時間軸方向に隣接するデータの間において対応するアドレスのコード間で差分をとり、上記対応するアドレスの差分どうしをまとめて新たなベクトルを複数生成するコード列再配列ステップと、

上記コード列再配列ステップで生成された複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化ステップとを備えたことを特徴とするデータ圧縮方法。

【請求項53】上記コード列再配列ステップで上記複数の新たなベクトルを生成する際に、上記新たなベクトル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようにしたことを特徴とする請求項50～52の何れか1項に記載のデータ圧縮方法。

【請求項54】少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータ列のうち、各時刻におけるデータ間で対応するアドレスのデータどうしをまとめて複数のベクトルとし、それらのベクトルに類似したコードベクトルをコードブックから夫々探し出し、それらに対応するコード列を出力するベクトル量子化ステップを有することを特徴とするデータ圧縮方法。

【請求項55】 上記複数のベクトルを取り出す際に、上記ベクトル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようすることを特徴とする請求項54に記載のデータ圧縮方法。

【請求項56】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

上記圧縮対象から少なくとも1つ以上の特徴量を抽出し、特徴量データと上記圧縮対象から上記特徴量を除いた基本パターンデータとに分離し、分離した特徴量データと基本パターンデータとに対してそれぞれ独立にベクトル量子化を行うようにしたことを特徴とするデータ圧縮方法。

【請求項57】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルをコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力するベクトル量子化ステップと、

上記ベクトル量子化ステップで各時刻毎に出力されるコード列について、時間軸方向に隣接するデータの対応するアドレスどうしでコード間の相関を夫々とり、相関が所定値より小さいアドレスのコードのみを出力する出力ステップとを有することを特徴とするデータ圧縮方法。

【請求項58】 少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象のカラー画像より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮方法において、

輝度信号用のコードブックと色信号用のコードブックとを備え、上記色信号用のコードブックよりも上記輝度信号用のコードブックの方により多くのコードベ

クトルを割り当てるようにしたことを特徴とするデータ圧縮方法。

【請求項59】 上記輝度信号であるY信号、および上記色信号であるU信号、V信号の圧縮レートが4：1：1もしくは4：2：2であることを特徴とする請求項58に記載のデータ圧縮方法。

【請求項60】 圧縮対象より抽出されるベクトルの特徴量と、上記コードブック内の各コードベクトルに関する特徴量とを比較し、その比較結果に基づいて上記コードベクトルについて上記圧縮対象ベクトルとの類似度を求める演算を省略するようにしたことを特徴とする請求項43～45、50～59の何れか1項に記載のデータ圧縮方法。

【請求項61】 あるコードベクトルの特徴量と上記圧縮対象ベクトルの特徴量との差分絶対値を求める差分絶対値演算ステップと、

上記差分絶対値演算ステップで求められた差分絶対値が、他のコードベクトルについて既に求められている類似度を表す値の最小値よりも大きい場合に、上記あるコードベクトルについての類似度の演算を省略する省略判定ステップとを備えることを特徴とする請求項60に記載のデータ圧縮方法。

【請求項62】 異なる種類の特徴量を複数利用し、上記複数種類の特徴量と上記圧縮対象のベクトルより抽出される複数種類の特徴量とに基づいて、上記各コードベクトルのそれぞれについて上記圧縮対象ベクトルとの類似度を求める演算を省略するかどうかを決定するようにしたことを特徴とする請求項60または61に記載のデータ圧縮方法。

【請求項63】 少なくとも1つ以上のコードブックを保持しておくコードブックサーバと、データ圧縮システムと、データ伸長システムとを備えたデータ圧縮伸長システムにおいて、上記コードブックサーバが、上記データ圧縮システムおよびデータ伸長システムからの要求に従って、保持している何れかのコードブックを供給するようにしたことを特徴とするデータ圧縮伸長方法。

【請求項64】 請求項1に記載の機能をコンピュータに実現させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項65】 請求項2、4、6の何れか1項に記載の各手段としてコンピュータを機能させるためのプログラムを記録したことを特徴とするコンピュー

タ読み取り可能な記録媒体。

【請求項66】 請求項9～25の何れか1項に記載の各手段としてコンピュータを機能させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項67】 請求項29～32の何れか1項に記載のコードブックの作成方法の処理手順をコンピュータに実行させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項68】 請求項34または40に記載の各手段としてコンピュータを機能させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項69】 請求項41に記載の機能をコンピュータに実現させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項70】 請求項50～52、54、57の何れか1項に記載の各ステップをコンピュータに実行させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項71】 請求項56または60に記載のデータ圧縮方法の処理手順をコンピュータに実行させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項72】 請求項63に記載のデータ圧縮伸長方法の処理手順をコンピュータに実行させるためのプログラムを記録したことを特徴とするコンピュータ読み取り可能な記録媒体。

#### 【発明の詳細な説明】

##### 【0001】

##### 【発明の属する技術分野】

本発明はデータ圧縮装置および方法、データ伸長装置および方法、データ圧縮伸長システムおよび方法、コードブックの作成方法、更にはこれらの処理を実行させるためのプログラムを記憶した記録媒体に関し、特に、データ圧縮手法の1つとしてベクトル量子化を用いる装置、システムおよび方法に用いて好適なもの

である。

### 【0002】

#### 【従来の技術】

従来、データ圧縮の手法が種々提案されている。その中で、圧縮データの伸長処理を非常に簡単に行うことが可能なデータ圧縮アルゴリズムの1つとして、「ベクトル量子化」という手法が良く知られている。このアルゴリズムは、古くから信号処理の分野で知られており、特に、画像信号や音声信号のデータ圧縮、あるいはパターン認識に応用されてきた。

### 【0003】

このベクトル量子化では、ある大きさ（例えば $4 \times 4$ 画素のブロック）の画素パターンを幾つか用意しておき、それぞれにユニークな番号などを与える（この集合体を「コードブック」という）。そして、例えば2次元配列の画像データ中から同じ大きさ（例えば $4 \times 4$ 画素）のブロックを順次取り出し、それと最も似通ったパターンをコードブック中から見つけ出して、そのパターンの番号を当該ブロックに当てはめるというデータ圧縮を行う。ベクトル量子化では、1つのブロック内のデータ列が1つのベクトルに対応する。

### 【0004】

このようにコード化された圧縮データの受信側あるいは伸長側では、各ブロック毎に番号に対応するパターンをコードブックの中から取り出すだけで、元の画像を再現することができる。したがって、伸長側では、コードブックさえ受け取っているか、あるいはあらかじめ保持していれば、特に特殊な演算は必要としないため、非常に簡単なハードウェアで元の画像を再生することが可能となる。

### 【0005】

#### 【発明が解決しようとする課題】

上述のようなベクトル量子化において、例えば画像や音声信号のデータ圧縮を行う際に、高い圧縮率を保持したままいかに高品質の再生画像や再生音声を得るのか、また、ベクトル量子化を実行する上で必ず必要となるコードブックといかに性能の良いものを作成するかが課題となっている。

### 【0006】

従来、高い圧縮率を実現するために、例えば、画像データから取り出すブロックの大きさを大きくしたり、コードブックを構成するブロックの個数を少なくするといった努力が成されている。また、コードブックの最適化の手法としては、Kohonen の自己組織化マップの手法などを始めとして幾つかの手法が用いられていた。

#### 【0007】

しかしながら、例えばベクトル量子化の手法を用いて画像をデータ圧縮する際に、高い圧縮率を実現しようとすると、どうしても再生画像の画質が劣化してしまうという問題が生じる。逆に、再生画像の画質を高めようとすると、ベクトル量子化以外の処理が入るため、結果的に圧縮率は高くならないという問題があった。

#### 【0008】

画像データの圧縮に用いられるベクトル量子化技術は、静止画に対してベクトル量子化を行うように成されている。したがって、動画像を圧縮する場合は、各々のフレームに対してベクトル量子化を行い、更にベクトル量子化以外の技術を組み合わせることにより実現されてきた。よって、特に動画の場合には、圧縮率が結果的に高くならない割には、再生画像の画質が悪いという問題があった。

#### 【0009】

また、上記Kohonen の自己組織化マップの手法などのコードブックの最適化技術は、サンプル画像などを用いて適當な数式処理を行うことにより、コードブックの最適化を図るものである。したがって、得られるコードブックは、最適化の際に使用したデータに対してのみ有用なコードブックとなってしまうという問題があった。

#### 【0010】

すなわち、例えば、ある人の顔の画像データを用いて最適化されたコードブックは、その最適化に用いた画像に対しては最良のコードブックとなるが、他の画像に対しては必ずしも最良のコードブックになるとは限らない。したがって、例えば、そのコードブックを他の人の顔の画像データに対して用いてデータ圧縮を実施すると、圧縮データから再生した画像の画質は低下することになる。

## 【0011】

さらに、最適化に用いた画像と同じ人の顔という分類に含まれる画像に対しては、再生画像として比較的良好な画質が得られても、風景や文字といった異なる分類の画像に対しては、画質が劣化してしまうことが多い。つまり、コードブックのパターンが画像によって全く異なっているため、汎用性の低いコードブックになってしまうという問題があった。

## 【0012】

また、コードブックを用いたベクトル量子化の技術では、入力ベクトルと最も似通ったパターンのコードベクトルをコードブック中から探し出す処理や、圧縮コードに対応するコードベクトルをコードブック中から取り出して表示する処理に多くの演算を要し、処理に時間がかかるという問題もあった。

## 【0013】

本発明は、このような問題を解決するために成されたものであり、例えば画像や音声信号のデータ圧縮をベクトル量子化によって行う場合に、データ圧縮を高い圧縮率で実現するとともに、それにより得られた圧縮データをもとに品質の高いデータを再生できるようにすることを目的とする。

また、本発明は、高性能で、種々の画像に対応できる汎用性の高いコードブックを実現できるようにすることも目的とする。

さらに、本発明は、ベクトル量子化による圧縮時および伸長による画像再生時の処理速度を向上させることも目的とする。

## 【0014】

## 【課題を解決するための手段】

第1の本発明は、少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、上記ベクトルを構成するブロックを、データ位置を1次元的に配列したラインブロックとすることを特徴とする。

## 【0015】

第2の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化し

てベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、上記圧縮対象のベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するようにするブロックシフト手段を備えたことを特徴とする。

## 【0016】

第3の発明は、少なくとも1つ以上のデータを有するデータ列をベクトルとし、少なくとも1つ以上のコードベクトルを有するコードブックの中から圧縮コードに対応するコードベクトルを探し出して、それを該当するブロック位置に割り当てるこことによって元データを再現するデータ伸長装置において、上記コードブックの中から探し出したコードベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして割り当てるブロックシフト手段を備えたことを特徴とする。

## 【0017】

第4の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、上記第1のベクトル量子化手段により各時刻毎に出力されたコード列を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とする。

## 【0018】

第5の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化し

てベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、ある時刻におけるデータのコード列を基準コード列とし、上記基準コード列と他の時刻におけるデータのコード列との間において対応するアドレスのコード間で差分をとった結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とする。

## 【0019】

第6の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルを第1のコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力する第1のベクトル量子化手段と、上記第1のベクトル量子化手段により各時刻毎に出力されたコード列のうち、時間軸方向に隣接するデータの間において対応するアドレスのコード間で差分をとった結果を再配列して複数の新たなベクトルとし、上記複数の新たなベクトルに対してそれぞれ、それらのベクトルに類似したコードベクトルを第2のコードブックから夫々探し出し、それらに対応するコード列を出力する第2のベクトル量子化手段とを備えたことを特徴とする。

## 【0020】

第7の発明は、上記複数の新たなベクトルを生成する際に、上記新たなベクト

ル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようとする時間軸シフト手段を備えたことを特徴とする。

また、上記第4～第6の発明であるデータ圧縮装置とは逆の処理を行う伸長側にて複数の新たなベクトルを生成する際に、上記新たなベクトル内の少なくとも近接する要素を時間軸方向にずらして配列するようとする時間軸シフト手段を備えたことを特徴とする。

#### 【0021】

第8の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、上記圧縮対象から少なくとも1つ以上の特徴量を抽出し、特徴量データと上記圧縮対象から上記特徴量を除いた基本パターンデータとに分離する分離手段を備え、上記分離した特徴量データと基本パターンデータとに対してそれぞれ独立にベクトル量子化を行うようにしたことを特徴とする。

#### 【0022】

第9の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれ、当該データ中から複数のブロックを抽出してそれらのベクトルに類似したコードベクトルをコードブックから夫々探し出し、それらに対応するコード列を各時刻毎に出力するベクトル量子化手段と、上記ベクトル量子化手段により各時刻毎に出力されるコード列について、時間軸方向に隣接するデータの対応するアドレスどうしでコード間の相関を夫々とり、相関が所定値より小さいアドレスのコードのみを出力する出力制御手段とを備えたことを特徴とする。

#### 【0023】

第10の発明は、少なくとも1つ以上のデータを有するデータ列をブロック化

してベクトルとし、あらかじめ用意されたコードブックの中から、圧縮対象のカラー画像より抽出されるベクトルに類似したコードベクトルを探し出して、それに対応するコードを出力するデータ圧縮装置において、輝度信号用のコードブックと色信号用のコードブックとを備え、上記色信号用のコードブックよりも上記輝度信号用のコードブックの方により多くのコードベクトルを割り当てるようにしたことを特徴とする。

## 【0024】

第11の発明は、少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化の処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、上記初期のコードブックとして、とり得る値の最小値から最大値までデータ値が連続的に変化するパターンのコードブックを用いたことを特徴とする。

また、上記1回の処理で更新する範囲を上記仮想的な2次元平面上で1次元的に適用し、かつ更新回数の増加と共に範囲を減少させることを特徴とする。

また、上記更新係数の初期値を0.3～1の範囲内の何れかの値とし、かつ更新回数の増加と共に上記更新係数の値を減少させることを特徴とする。

## 【0025】

第12の発明は、少なくとも1つ以上のデータを有するデータ列であるベクトルの集合から成り、ベクトル量子化で用いられるコードブックを作成する方法であって、あるサンプルデータと初期のコードブックとを用いてベクトル量子化の処理を繰り返し行い、コードブックの仮想的な2次元平面上で指定された範囲の内容を1回の処理毎に所定の更新係数に従って更新していくことによってコードブックを最適化するコードブックの作成方法において、複数のサンプルデータに対して独立にコードブックを最適化し、得られた複数のコードブックを合成して新たなコードブックを作成するようにしたことを特徴とする。

## 【0026】

第13の発明は、コードブック内の各コードベクトルに関してあらかじめ求められた特徴量を記憶する特徴量記憶手段と、上記圧縮対象より抽出されるベクトルの特徴量を求める特徴量演算手段と、上記特徴量記憶手段に記憶されている各コードベクトルの特徴量と上記特徴量演算手段により求められた圧縮対象ベクトルの特徴量とに基づいて、上記各コードベクトルのそれぞれについて上記圧縮対象ベクトルとの類似度を求める演算を省略するかどうかを決定する演算省略手段とを備えたことを特徴とする。

#### 【0027】

第14の発明は、少なくとも1つ以上のコードブックを保持しておくコードブックサーバと、データ圧縮システムと、データ伸長システムとを備えたデータ圧縮伸長システムであって、上記コードブックサーバは、上記データ圧縮システムおよびデータ伸長システムからの要求に従って、保持している何れかのコードブックを供給することを特徴とする。

#### 【0028】

##### 【発明の実施の形態】

###### (第1の実施形態)

第1の実施形態においては、静止画に対してベクトル量子化を行う際に、通常は例えば $4 \times 4$ 画素のような正方形のブロックに対してベクトル量子化を行うところを、 $16 \times 1$ 画素のように1ラインで成るブロックに対してベクトル量子化を行う。さらに、ベクトル量子化時あるいはその圧縮画像の再生の際に、画像の水平方向にブロックを各ライン毎にずらす処理を行う。

#### 【0029】

図1および図2は、それぞれ第1の実施形態に係るデータ圧縮装置および伸長装置の構成を示す機能ブロック図であり、図3および図4は、それぞれ図1および図2に示したデータ圧縮装置およびデータ伸長装置の動作を示すフローチャートである。また、図5および図6は、本実施形態による伸長側の動作を説明するための説明図である。

#### 【0030】

まず、図5および図6について説明する。図5では、ベクトルとして画像上の

走査線に沿ったラインブロックをベクトルデータとして用いた場合（本実施形態のラインブロック方式）について、従来の正方形のブロックをベクトルデータとした場合（マクロブロック方式）と比較して説明している。図5（a）は従来のマクロブロック方式を示し、図5（b）は本実施形態のラインブロック方式を示している。

#### 【0031】

図5（a）に示すように、コードブックメモリには、例えば $4 \times 4$ 画素のマクロブロックを構成する16個のデータ列を1つのコードベクトルとして、このコードベクトルがアドレス順に多数記憶されている。このコードベクトルを利用して再生画像を生成し、それをディスプレイ等に表示する際には、16個のデータ列が $4 \times 4$ の正しい画素位置に表示されるように、ディスプレイメモリ上でアドレス変換を行う必要があった。

#### 【0032】

これに対して、図5（b）に示す本実施形態のラインブロック方式では、 $16 \times 1$ 画素の1つのライン上でブロックを抽出してベクトル量子化を行っている。このラインブロック内のデータ配列は、ディスプレイのスキャン方向と合致しているので、正方形のマクロブロックの場合に必要だったアドレス変換操作は回避することができる。また、この場合のコードブックも、Kohonen の自己組織化マップの手法を用いれば容易に得ることができる。

#### 【0033】

図7は、 $640 \times 480$ 画素の1枚の画像を圧縮、伸長、表示する処理を行う際に、それぞれの処理にかかる所用時間を測定した結果を示す図である。この図7を見ると、従来のマクロブロック方式では、ベクトル量子化（VQ）に約50 msec、デコードに約160 msec、表示に約60 msecの時間を要している。デコードにこれだけの時間がかかる理由は、上述したように、コード番号で指定されたコードベクトルをコードブックメモリから取り出してディスプレイに表示できる形式にする際に、アドレス変換が必要なためである。

#### 【0034】

このような場合、一般的にはアドレス変換用の専用チップを用意してその高速

化を図るのだが、本実施形態ではもっと簡単に高速化を図るための手段として、上述のラインブロック方式による手法を提案している。

図7には、ラインブロック方式をとった場合の各処理時間も示している。ベクトル量子化および表示走査にかかる時間はマクロブロック方式の場合と変化はないが、デコード時間は160 msecから大幅に短縮できていることが分かる。

#### 【0035】

また、マクロブロック方式およびラインブロック方式の何れの場合も（特にラインブロック方式をとった場合）、コードベクトルのパターン画像を順番に整然と貼り込んで再生画像を表示すると、再生画像の各ラインの両端のパターンの大きな不連続点がどうしても目立ってしまう。そこで、図6に示すように、本実施形態では、コードベクトルのパターンを貼り込む位置を各ライン毎に少しずつずらす工夫をとった（以下、これをスライドブロック方式と称する）。

#### 【0036】

このように、少なくとも近接するライン間ではコードベクトルのブロックを画像の水平方向にずらしてはめ込むようにすると、各ブロック境界の連續性が断たれてブロック境界（量子化誤差）が目立ちにくくなる。これにより、再生画像の両端に毛羽立ったような部分は見えるが、概ね良好な画像が得られるようになる。なお、この画像の両端部分の乱れは、表示をするときにマスクをかけることで容易に除去することが可能である。

#### 【0037】

本実施形態では、近接するライン間で画像の水平方向にブロックをずらしていく処理を、図6で説明したようにベクトル量子化後の画像の再生時に行うだけでなく、ベクトル量子化時にも行う。すなわち、画像の左上から順番にきちんとブロックを切り出してくるのではなく、画像の水平方向にずらしながらブロックを切り出してベクトル量子化を行うようにしている。

#### 【0038】

次に、以上に述べた第1の実施形態に係る構成および動作を、主に図1～図4に基づいて説明する。図1において、1は画像入力部であり、圧縮対象の原画像データを入力する。入力された画像データは、画像メモリ2に一旦蓄えられ、読

み出し部3からのアドレス制御等により読み出されて出力される。

#### 【0039】

4はブロック化部であり、上記画像メモリ2から読み出された画像データを、例えば $16 \times 1$ 画素単位のラインブロックに分割する。その際、少なくとも近接するライン間で画像の水平方向にずらしながらブロックを切り出すようとする。なお、近接する水平ラインとの間でブロックの位置がずれていれば、ずらす方向と量は任意に設定可能である。また、ここではブロック化部4を設けているが、読み出し部3が画像データの読み出しを行うときに、上述のようにラインプロックをずらしながら読み出すようにすれば、このブロック化部4は必ずしも必要ではない。

#### 【0040】

5はコードブック記憶部であり、原画像から抽出されるラインブロックと同じ大きさ（ $16 \times 1$ 画素のブロック）の画素パターンをコードベクトルとしてあらかじめ複数記憶している。6はベクトル量子化部であり、上記ブロック化部4によって原画像から抽出されたある1つのラインブロックのデータと、コードブック記憶部5に記憶されている複数のコードベクトルのデータとを夫々比較することにより、原画像のブロックデータと最も似通ったパターンをコードブック中から見つけ出して、そのパターンのコード番号を出力する。この操作を、原画像中から抽出される全てのラインブロックについて行う。

#### 【0041】

また、図2において、7はコード番号入力部であり、圧縮側で特定された各ブロック毎のコード番号列を入力する。入力されたコード番号列は、コードメモリ8に一旦蓄えられ、読み出し部9からのアドレス制御等により読み出されて出力される。10はコードブック記憶部であり、例えば $16 \times 1$ 画素で成るブロックの画素パターンをコードベクトルとしてあらかじめ複数記憶している。このコードブックは、伸長処理に先立って圧縮側のコードブックを伝送して記憶するようにしても良いし、最初から同じものを記憶しておくようにしても良い。

#### 【0042】

11は再生画像生成部であり、コードメモリ8より読み出されたコード番号列

をもとに、これらに対応するコードベクトルのパターン画像を夫々コードブック記憶部10から読み出して当てはめる処理を行う。12はブロックシフト部であり、再生画像生成部11より出力された再生画像を構成する各ブロックを画像の水平方向にずらす処理を行う。その際、例えば少なくとも近接する水平ライン間でブロックの位置がずれていれば、ずらす方向と量は任意に設定可能である。このようなシフト処理を施された再生画像は、画像メモリ13に一旦蓄積された後、図示しない表示装置に送られて表示される。

#### 【0043】

上記図1に示した圧縮側の動作を示す図3において、まずステップS1で初期化処理を行う。ここでは、処理済のブロックをカウントするためのブロックカウンタ（図1では図示せず）、画像バッファ（図1の画像メモリ2）、およびベクトル量子化により決定したコード番号を格納するためのコードバッファ（図1では図示せず）をそれぞれクリアする処理を行う。

#### 【0044】

次に、ステップS2で原画像を入力し、画像バッファに格納する。そして、ステップS3で、その画像バッファに格納された原画像中から $x \times y$ 画素の大きさのブロック（ $x, y$ は任意の値だが、上述の例では $x = 16, y = 1$ ）を読み出す。この読み出しの際には、少なくとも近接する水平ライン間でブロックの位置がずれるようにブロックの位置を調整する。

#### 【0045】

そして、ステップS4で、読み出したブロックのデータに対してベクトル量子化の処理を行い、コードブック記憶部5に記憶されているコードブックの中から上記読み出したブロックのデータと最も相関度の大きいコードベクトルを探し出し、それに対応するコード番号（Winner code）を決定する。次に、ステップS5で、得られたコード番号をコードバッファに格納する。ここに格納されたコード番号は、その後外部に出力される。なお、このコードバッファは必ずしも必要ではない。

#### 【0046】

次に、ステップS6に進み、画像中の全てのブロックについて上述の処理が終

了したかどうかを判断する。終了していなければ、ステップS7でブロックカウンタの値を1だけ増やした後、ステップS3に戻って同様の処理を繰り返す。このようにして画像中の全てのブロックについてコード番号を決定して出力することにより、原画像が圧縮される。

#### 【0047】

また、上記図2に示した伸長側の動作を示す図4において、まずステップS8で初期化処理を行う。ここでは、処理済のブロックをカウントするためのブロックカウンタ（図2では図示せず）、再現画像バッファ（図2の画像メモリ13）、および入力されたコード番号を格納するためのコードバッファ（図2のコードメモリ8）をそれぞれクリアする処理を行う。

#### 【0048】

次に、ステップS9で、圧縮側で生成されたコード番号列を入力し、コードバッファに格納する。そして、ステップS10で、格納されたコード番号列のうち、現在のブロックカウンタで示される1つのコード番号をもとにデコード処を行なう。すなわち、コードブック記憶部10に記憶されているコードブックの中から、コード番号に対応するコードベクトル（パターン画像）を取り出して再現画像バッファに格納する。その際、取り出された $x \times y$ 画素のブロックを、少なくとも近接する水平ライン間で位置がずれるように格納する。

#### 【0049】

ここで、本実施形態では、圧縮側のステップS3と伸長側のステップS10とで、ブロックのずらし方が互いに逆の関係となるようとする。すなわち、ステップS3においてあるブロックラインについて右方向にi画素分ずらしたら、ステップS10では、そのブロックラインについては左方向にi画素分ずらすようとする。このようにすることで、再生される再現画像中には各ブロックライン毎のずれがなくなくなるので、量子化誤差を目立たなくすることができるだけでなく、再生画像の画質を更に向上させることができる。

#### 【0050】

次に、ステップS11に進み、画像中の全てのブロックについて上述の処理が終了したかどうかを判断する。終了していなければ、ステップS12でブロック

カウンタの値を1だけ増やした後、ステップS10に戻って同様の処理を繰り返す。このようにして画像中の全てのブロックについてコードブック中からパターン画像を取り出して再現画像バッファに格納することにより、ここに格納された再現画像がディスプレイ等に与えられて表示される。

#### 【0051】

なお、以上の例では、圧縮側と伸長側との両方でブロックをずらす例を説明したが、圧縮側においてのみ、もしくは伸長側においてのみブロックをずらすようにもしても良い。この場合は、再現画像中に各ブロックライン毎のずれが含まれることはあるものの、量子化誤差自体は目立たなくすることができ、従来例に比べて再生画像の画質を向上させることができる。

#### 【0052】

また、ここでは静止画のデータ圧縮について説明しているが、動画の場合は個々のフレームに対してベクトル量子化を行うことから、この動画のデータ圧縮についても同様に適用することができる。

本実施形態では、図8に示すベクトル量子化PCIバスボード22を用いて、動画像をベクトル量子化により圧縮し、圧縮を行った結果から動画像の再生を行うシステムを実現している。以下に、その詳細な説明を行う。

#### 【0053】

図8に示すベクトル量子化ボード22には、例えば8ビット16次元( $4 \times 4$ 画素)のコードベクトルを256個搭載可能なチップが8個実装されており、合わせて2048個のコードベクトルを用いてベクトル量子化を実行可能となっている。これらのチップは、例えばFPGA(Field Programmable Gate Array)により構成されたPCIバスインターフェースを介して図示しないコンピュータと接続されている。

#### 【0054】

ベクトル量子化を行う際には、まずあらかじめ作成しておいたコードブックを構成している全てのコードベクトルをメモリ21からベクトル量子化ボード22のチップ上に転送しておく。次に、あらかじめメモリ21に保存しておいた動画像データを順次ベクトル量子化ボード22に送り、ベクトル量子化を実行する。

ベクトル量子化の結果得られたコード番号は、一旦メモリ21に格納される。そして、このコード番号をもとにそれに対応するコードベクトルをメモリ21から読み出し、デコードを実行する。デコードされた再生画像データは、コンピュータのディスプレイ23に送られ、画像として表示される。

#### 【0055】

ここでは、コードブックを構成しているコードベクトルの個数を2048個とし、コードベクトルを構成しているデータの個数を16個としているが、これは1つの具体例であり、ここで述べた数値に制限されるものでないことは言うまでもない。また、ここで構成したシステムにおいて用いた画像サイズ、並びにベクトル量子化を行う際のブロックサイズについても、1つの具体例であり、ここで述べた数値に制限されるものでないことも言うまでもない。

#### 【0056】

以上のように、第1の実施形態では、ベクトル量子化の処理単位であるブロックを、例えば $16 \times 1$ 画素のようなラインブロックとしている。これは、ディスプレイのスキャン方向に合っているため、再生の際のアドレス変換処理が不要となり、画像の再現スピードが速くなる（ $4 \times 4$ 画素のマクロブロックでベクトル量子化を行った場合に比べて、約3.5倍の再現スピードが得られる）。

#### 【0057】

例えば、1フレームが $640 \times 480$ 画素の動画像に対してマクロブロック方式によりデータ圧縮を行うと、フレームレートは4フレーム/秒程度であるが、本実施形態のラインブロック方式によれば、毎秒14フレーム程度のベクトル量子化動作が実現できる。

#### 【0058】

表示部分の処理時間の短縮は、表示プログラムの改善により実現可能である。また、現在のシステムでは、ベクトル量子化処理にかかる時間は、その6割程度が入力ベクトルをベクトル量子化ボードに転送するためのアドレス変換および転送時間で占められており、実際にベクトル量子化チップが動作しているのはわずかである。したがって、転送バス幅を拡張してCPUの処理能力を高めることにより、処理速度を更に短縮することが可能である。これらの改善を実行すること

で、毎秒30フレームのベクトル量子化動作は実現可能である。

#### 【0059】

また、本実施形態では、画像の水平方向に各ブロックをずらしながら切り出してベクトル量子化を行い、あるいは各ブロックをずらしながら画像の再現を行っているので、ブロックをそのまま順番にはめ込んだ場合に画像の両端において目立つブロック境界による縦線（量子化誤差）が分散されて目立たなくなり、画質を向上させることができる。

#### 【0060】

なお、以上の実施形態ではラインブロック方式とスライドブロック方式とを併用しているが、何れか一方のみを用いても良い。ラインブロック方式だけを採用した場合は少なくとも画像の再現スピードを速くすることができ、スライドブロック方式だけを採用した場合は少なくとも量子化誤差を目立たなくすることができます。

#### 【0061】

また、以上の実施形態では、1枚の画像内で空間方向に対してブロックをずらす例を示しているが、複数のフレームを有する動画像をデータ圧縮する場合において、フレーム間において時間軸方向に対してブロック内の要素をずらすようにしても良い。このようにした場合にも、再生画像中の量子化誤差を目立たなくすることができる。

#### 【0062】

##### (第2の実施形態)

動画像に対して圧縮を行う際に、動画を構成する各々のフレーム（静止画）に対してベクトル量子化を行うだけでは、高い圧縮率を得ることは難しい。そこで第2の実施形態では、複数フレームの各静止画に対してフレーム内でのベクトル量子化を行うだけでなく、複数フレームの各静止画の同じアドレスどうしのデータをベクトルと成し、フレーム間でもベクトル量子化を適用してデータ圧縮を行うようとする。

#### 【0063】

図9および図10は、第2の実施形態によるデータ圧縮伸長システムの一構成

例を示す機能ブロック図であり、図11は、第2の実施形態に係る圧縮時のデータの流れを示すデータフロー図である。なお、図9は圧縮側の構成を示し、図10は伸長側の構成を示している。また、伸長時のデータの流れは基本的に圧縮時の流れと逆であり、図11から想定できるので、伸長時のデータフロー図はここでは特に示していない。

#### 【0064】

ここでは、図11に示すように、1フレームが $640 \times 480$ 画素からなり、1画素当たりの輝度値が8ビット表現された動画像に対して圧縮伸長を行う場合について述べる。また、ここでは説明の都合上、16フレーム分の画像に対して圧縮伸長を行っているものとする。

#### 【0065】

図9において、31は画像入力部であり、圧縮対象の原画像データ、ここでは動画像データを各フレーム毎に順次入力する。入力された画像データは、各フレーム毎に画像メモリ32に一旦蓄えられ、読み出し部33からのアドレス制御等により読み出されて出力される。

#### 【0066】

34はブロック化部であり、上記画像メモリ32から読み出された画像データを、例えば $4 \times 4$ 画素単位のブロックに分割する。なお、ここではブロック化部34を設けているが、読み出し部33が画像データの読み出しを行うときに、ブロック単位で読み出すようにすれば、このブロック化部34は必ずしも必要ではない。

#### 【0067】

35は空間方向用コードブック記憶部であり、1枚の静止画（フレーム）に対して当該フレーム内でベクトル量子化を行うためのコードブック（1つのフレーム空間内でベクトル量子化を行うためのコードブックであるので、以下ではこれを「空間方向用コードブック」と称する）をデータベースとしてあらかじめ記憶しておくものである。

#### 【0068】

ここでは、この空間方向用コードブックは、16個のデータからなるデータ列

を1つのコードベクトルとし、そのコードベクトルを複数組み合わせたものである。ここでは、2048ベクトル分を組み合わせて空間方向用コードブックを構成している。

#### 【0069】

36は相関度演算部であり、演算モード指定部37により空間方向ベクトル量子化を行うモードが指定されているときは、順次入力される各フレーム毎に、上記空間方向用コードブック記憶部35に記憶されている空間方向用コードブックを用いて、当該コードブック中の複数のコードベクトルと原画像からの入力ベクトルとの相関、つまり類似度を算出する。

#### 【0070】

ここで類似度とは、例えば、2つのベクトルデータがある関数に入力して演算することで、両者がどれくらい似ているかを数値化したものと言う。上記関数の代表的なものとしては、2つの入力ベクトルデータのマンハッタン距離（差分絶対値距離）やユークリッド距離を求める関数が挙げられる。例えばマンハッタン距離は、入力ベクトルを構成している個々の要素のデータと、コードベクトルを構成してい個々の要素のデータとの差分絶対値を夫々求め、得られたそれぞれの差分絶対値を加算することで得られる。

#### 【0071】

入力ベクトルに最も似通ったコードベクトルを探し出すためには、空間方向用コードブックを構成する各コードベクトル毎に例えば上述のようなマンハッタン距離を求め、その距離が最も小さいものを探し出す。

なお、類似度を求めるための関数は、上記の関数に限らず、目的に応じて任意の関数を使うことができる。例えば、一方あるいは双方のベクトルデータに対して係数（重み）を付けた関数や、ブロック内の特徴量データ（例えば、ベクトルデータの要素の総和）を計算する関数などを使っても良い。

#### 【0072】

38はコード決定部であり、上記相関度演算部36による類似度の演算結果に基づいて、類似度の最も大きいコードベクトルに対応するコード番号をそのブロックの圧縮コード（Winner code）として出力する。この相関度演算部36およ

びコード決定部38によりベクトル量子化手段が構成される。ここでベクトル量子化の手法としては、従来の手法を用いても良いし、第1の実施形態で述べた手法を用いても良い。

#### 【0073】

以下、この空間方向ベクトル量子化の動作について、図11を用いて詳しく説明する。まず、第1フレーム目の画像について、上記空間方向用コードブックを用いてベクトル量子化を行う。ここでは、第1フレーム目の画像を水平方向に4画素、垂直方向に4画素ずつ区切ったブロック領域をそれぞれ入力ベクトルと定義し、その1つの入力ベクトルと、空間方向用コードブックを構成している各コードベクトルとの類似度を求め、入力ベクトルに最も似通ったコードベクトルのコード番号を特定する。

#### 【0074】

この操作をフレーム内の全ての入力ベクトルに対して行うことにより、第1フレーム目の画像を $4 \times 4$ 画素に分割した各ブロック領域を、空間方向用コードブック中のコードベクトルに対応するコード番号で置き換える。図11中に127, 198, …と示した数字は、画像の左上から右下方向に向かって順に抽出したそれぞれのブロックについて置き換えたコード番号を示している。

#### 【0075】

次に、第2フレーム目の画像から第16フレーム目の画像まで計15枚の画像に対して、上記第1フレーム目の画像と同じ処理を行い、それぞれのフレーム画像について、フレーム内を夫々 $4 \times 4$ 画素に分割した各ブロック領域を、空間方向用コードブック中のコードベクトルに対応するコード番号で置き換える。

#### 【0076】

図9に戻り、39は空間方向コード再配列演算部であり、上記空間方向ベクトル量子化手段によって空間方向用コードブックのコード番号で表現された各フレームの画像に対して、同じアドレスが示すブロックのコード番号をフレーム画像順に並べ替えることにより、時間軸方向に分散して存在する同アドレスブロックの各コード番号を1つのデータ列としてまとめて表現する。

#### 【0077】

つまり、図11のように16フレーム分の画像に対して空間方向ベクトル量子化を行った場合、第1フレーム目の画像中の指定されたアドレスのブロックのコード番号が、新しく作られるデータ列の0番地の位置に置かれ、第2フレーム目の画像中の上記第1フレーム目の画像と同じアドレスのブロックのコード番号が、新しく作られるデータ列の1番地の位置に置かれる。このように、16フレーム分の同アドレスのブロックのコード番号に対して並べ替えが行われ、最後の第16フレーム目の画像中の同じアドレスのブロックのコード番号は、新しく作られるデータ列の15番地の位置に置かれることになる。

#### 【0078】

このようにして新しく生成された16個のコード番号列を含む集合を、新たなベクトルとする。以下では、これを「時間軸ベクトル」と称する。図11に示した例の場合、例えば、第1フレーム目～第16フレーム目の各フレームの先頭アドレスで指定されるブロックのコード番号を並べ替えた(127, 287, 58, …, 1483, 876)のデータ列が、1つの時間軸ベクトルに相当する。

#### 【0079】

本例のように、1フレームが $640 \times 480$ 画素の画像の場合で、かつ1つのブロック領域を $4 \times 4$ 画素とした場合、1フレームの画像からは $160 \times 120 = 19200$ 個のブロックが生成される。よって、上述のような並べ替え処理を空間方向ベクトル量子化を行った画像のすべての範囲に対して行うと、19200個の時間軸ベクトルが新しく生成されることとなる。

#### 【0080】

再び図9に戻り、40は時間軸用コードブック記憶部であり、上記のように作成された時間軸ベクトルに対してベクトル量子化を行うためのコードブック（時間の経過と共に与えられる複数のフレーム間に渡るデータに対してベクトル量子化を行うためのコードブックであるので、以下ではこれを「時間軸用コードブック」と称する）をデータベースとしてあらかじめ記憶しておく。この時間軸用コードブックも、16個のデータからなるデータ列を1つのコードベクトルとし、そのコードベクトルを2048個組み合わせたものである。

#### 【0081】

上記空間方向コード再配列演算部39により画像のすべての範囲に対して並べ替えが行われると、時間軸ベクトル量子化開始信号が出力され、それが演算モード指定部37に与えられる。これに応じて演算モード指定部37は、時間軸ベクトル量子化を行うモードに切り替える。相関度演算部36は、演算モード指定部37により時間軸ベクトル量子化を行うモードが指定されているときは、上記空間方向コード再配列演算部39により作成された各時間軸ベクトルに対して、上記時間軸用コードブック記憶部40に記憶されている時間軸用コードブックを用いてベクトル量子化を行う。

#### 【0082】

以下、この時間軸ベクトル量子化の動作について、図11を用いて詳しく説明する。すなわち、先に生成された19200個の時間軸ベクトルのうち、ある1つの時間軸ベクトルと、時間軸用コードブックを構成している各コードベクトルとの類似度を演算し、その類似度が最も大きいコードベクトルの時間軸用コードブック内のアドレス（コード番号）を特定する。なお、類似度の演算については上述した通りである。

#### 【0083】

この操作を全ての時間軸ベクトルに対して行うことにより、空間方向ベクトル量子化後のコード番号列で表現されていた複数の時間軸ベクトルを、時間軸用コードブック中の各コードベクトルに対応するコード番号列で置き換える。図11の例では、例えば上述した（127, 287, 58, …, 1483, 876）の時間軸ベクトルが“10”という時間軸コード番号に置き換えられている。この段階で、16フレーム分の $640 \times 480$ 画素分の画像は、例えば11ビット表現された19200個のデータから成るデータ列（10, 984, …）で表現されており、これが伸長側に供給される。

#### 【0084】

したがって、16フレーム分の動画像を圧縮側から伸長側に転送しようとする場合、空間方向用コードブックと時間軸用コードブックとをあらかじめ伸長側に送っておき（伸長側において最初から同じものを持っていても良い）、その後、上述の一連の処理によって得られた時間軸コードブックのコード番号列で表現さ

れたリストを送れば良い。これにより、39321600ビットの情報量を有する16フレーム分の画像を、約0.01倍にあたる422400ビットの情報量を有する時間軸コードブックのコード番号列にまで、画質を保持したまま圧縮することができる。

#### 【0085】

図10に示す伸長側では、このようにして圧縮側から転送されてきたデータをもとに、もとの16フレーム分の画像を再現することができる。まず、時間軸コード入力部41は、圧縮側より転送されてきた時間軸用コードブックのコード番号列を入力する。入力された時間軸コード番号列は、時間軸コードメモリ42に一旦蓄えられ、読み出し部43からのアドレス制御等により読み出されて出力される。

#### 【0086】

そして、空間方向コード再生部44は、時間軸コードメモリ42より読み出された時間軸コード番号列を用いて、個々の時間軸コード番号に対応するコードベクトルを、例えば圧縮側からあらかじめ転送され時間軸用コードブック記憶部45に記憶されている時間軸コードブックから取り出す。そして、取り出した各コードベクトルを順に配列することにより、空間方向のコード番号を再生する。

#### 【0087】

ここでは、時間軸コードメモリ42より読み出された時間軸コード番号列には19200個の時間軸コード番号が含まれており、これらの時間軸コード番号に対応する時間軸コードブックの1つのコードベクトルは16個のデータから構成されている。そのため、上記空間方向コード再生部44の処理により順番に配列されたコードベクトルのデータは、全部で307200個のデータ（空間方向コード番号）から構成される。

#### 【0088】

このようにして再生された空間方向コード番号列は、空間方向コードメモリ46に一旦蓄えられ、読み出し部47からのアドレス制御等により読み出されて出力される。次に、再生画像生成部48は、上記のように配列された307200個のデータ列から16個おきにデータを取り出し、19200個のデータで1つ

のデータ列を生成し、これをベクトルとする。以下ではこれを、上述した時間軸ベクトルに対して「空間方向ベクトル」と称する。

#### 【0089】

この操作を307200個のデータ列の全空間に渡って行うことにより、空間方向コードメモリ46から読み出した空間方向コード番号列を16個の空間方向ベクトルに16分割する。ここで生成された16個の空間方向ベクトルは、それぞれ第1フレーム目から第16フレーム目までの画像を再現するための空間方向コード番号列の集合である。

#### 【0090】

再生画像生成部48は、さらに、16個に分割された空間方向ベクトル列のそれぞれについて、その空間方向ベクトルを構成する個々の空間方向コード番号（19200個）に対応するコードベクトルを、例えば圧縮側からあらかじめ転送され空間方向用コードブック記憶部49に記憶されている空間方向用コードブックから参照して探し出す。そして、該当するコードベクトル（パターン画像）のデータを各ブロック位置に順次はめ込むことにより、元の画像を再生する。

#### 【0091】

このような処理を16個に分割された空間方向ベクトル列が得られる毎に行うことにより、元の16フレーム分の画像を再生することができる。なお、307200個の空間方向コード番号列から16個おきにデータを取り出す操作を読み出し部47が行うようにすれば、再生画像生成部48は、19200個のデータが読み出される毎に各フレームの再生画像を順に生成することができる。

#### 【0092】

このようにして再生された画像は、画像メモリ50に一旦蓄積された後、図示しない表示装置に送られて表示されたり、あるいは図示しない記憶装置に送られて記憶されたりする。

#### 【0093】

なお、この第2の実施形態では、1つのベクトルを構成するデータの数が16個であり、空間方向用コードブックおよび時間軸用コードブックがそれぞれ2048個のコードベクトルで構成されているとしたが、これは本実施形態を説明す

るにあたって分かりやすくするために挙げた例であり、ここで述べた数値に限定されるべきでないのは言うまでもない。

#### 【0094】

また、ここでは16フレーム分の画像をベクトル量子化により圧縮することについて述べたが、圧縮を行うフレームの枚数は16フレームに限定されるものではなく、必要な枚数のフレームをベクトル量子化により圧縮しても良いことは言うまでもない。また、ここでは動画像を用いた実施形態を示しているが、動画像に限らず、音声信号を対象にしても同様に適用することができる。

#### 【0095】

また、この第2の実施形態では、1つの相関度演算部36を空間方向ベクトル量子化と時間軸ベクトル量子化とで使い回すようにしているが、それぞれのベクトル量子化用に2つ設けても良い。

#### 【0096】

##### (第3の実施形態)

図12は、第3の実施形態に係る圧縮時のデータの流れを示すデータフロー図である。なお、第3の実施形態によるデータ圧縮伸長システムの構成は、第2の実施形態とほぼ同様なのでここでは図示を省略する。ただし、空間方向コード再配列演算部39および再生画像生成部48における処理内容が第3の実施形態では多少異なっているので、これについては詳しく説明する。

#### 【0097】

本実施形態では、図12に示すように、17フレーム分の画像に対して圧縮を行っている。このため、相関度演算部36では、演算モード指定部37により空間方向ベクトル量子化を行うことが指定されているときに、第1のフレーム目の画像から第17のフレーム目の画像まで計17枚の画像に対して空間方向ベクトル量子化の処理を行うことにより、それぞれのフレーム画像について、夫々 $4 \times 4$ 画素に分割した各ブロック領域を、空間方向用コードブック中のコードベクトルに対応するコード番号で置き換える。

#### 【0098】

空間方向コード再配列演算部39は、上記相関度演算部36によって空間方向

用コードブックのコード番号で表現された各フレームの画像中で同じアドレスが示すブロックのコード番号について、第1フレーム目（これを基準フレームとする）のコード番号と各フレームのコード番号との差分を夫々算出し、その結果をフレーム画像順に並べ替える。なお、並べ替えた後に個々のアドレス毎に基準フレームとの差分を計算しても良い。

#### 【0099】

すなわち、図12に示すように、まず第1フレーム目と第2フレーム目とで同じアドレスどうしのコード番号間で差分をとり、次に第1フレーム目と第3フレーム目とで同じアドレスどうしのコード番号間で差分をとる。以下同様にして、第1フレーム目と第4～第17フレーム目とで同じアドレスどうしのコード番号間で差分をとる。そして、同じアドレスが示すブロックの差分結果をフレーム画像順に並べ替えることにより、時間軸方向に分散して存在する同アドレスブロックの各差分コード番号を1つのデータ列としてまとめて表現する。

#### 【0100】

つまり、図12のように17フレーム分の画像に対して空間方向ベクトル量子化を行った場合、第1フレーム目の画像のコード番号と第2フレーム目の画像のコード番号との差が、新しく作られるデータ列の0番地の位置に置かれ、第1フレーム目の画像のコード番号と第3フレーム目の画像のコード番号との差が、新しく作られるデータ列の1番地の位置に置かれる。

#### 【0101】

このように、第1フレーム目のコード番号と残り16フレーム分のコード番号とのそれぞれの差に対して並べ替えが行われ、最後の第1フレーム目の画像のコード番号と第17フレーム目の画像のコード番号との差は、新しく作られるデータ列の15番地の位置に置かれることになる。相関度演算部36は、演算モード指定部37により時間軸ベクトル量子化を行うことが指定されているときは、このようにして並べ替えられた時間軸ベクトル列に対して時間軸ベクトル量子化の処理を行い、時間軸用コードブックの時間軸コード番号列を出力する。

#### 【0102】

この段階で、17フレーム分の $640 \times 480$ 画素分の画像は、例えば11ビ

ット表現された19200個のデータから成るデータ列で表現されている。したがって、17フレーム分の動画像を転送する場合、空間方向用コードブックと時間軸用コードブックとをあらかじめ伸長側に送っておき、その後、上述の一連の処理によって得られた時間軸コードブックのコード番号列で表現されたリストと、第1フレーム目の画像が空間方向用コードブックのコード番号列で表現されたデータ列とを送れば良い。

#### 【0103】

これにより、41779200ビットの情報量を有する17フレーム分の画像を、約0.01倍にあたる422400ビットの情報量を有する時間軸コードブックのコード番号列にまで、画質を保持したまま圧縮することができる。

#### 【0104】

伸長側では、このようにして圧縮側から転送されてきたデータをもとに、もとの17フレーム分の画像を再現することができる。再生画像生成部48によって307200個の配列データを16個の空間方向ベクトルに16分割するところまでは、第2の実施形態と同様である。本実施形態では、さらに、圧縮側より送られてきている第1フレーム目の空間方向ベクトル量子化後のコード番号と、再生画像生成部48により再生された各空間方向ベクトルを構成しているコード番号差分との加算を行い、コード番号のデータ列を再構成する。

#### 【0105】

再生画像生成部48は、第1フレーム目の空間方向ベクトルと、16個に分割され再構成された空間方向ベクトル列とのそれぞれについて、その空間方向ベクトルを構成する個々のコード番号に対応するコードベクトルを、空間方向用コードブック記憶部49に記憶されている空間方向用コードブックから探し出してはめ込んでいくことにより、元の画像を再生する。これにより、元の17フレーム分の画像を再生することができる。

#### 【0106】

なお、この第3の実施形態では、1つのベクトルを構成するデータの数が16個であり、空間方向用コードブックおよび時間軸用コードブックがそれぞれ2048個のベクトルで構成されているとしたが、これは本実施形態を説明するにあ

たって分かりやすくするために挙げた例であり、ここで述べた数値に限定されるべきでないのは言うまでもない。

#### 【0107】

また、ここでは17フレーム分の画像をベクトル量子化により圧縮することについて述べたが、圧縮を行うフレームの枚数は17フレームに限定されるものではなく、必要な枚数のフレームをベクトル量子化により圧縮しても良いことは言うまでもない。また、ここでは動画像を用いた実施形態を示しているが、動画像に限らず、音声信号を対象にしても同様に適用することができる。

#### 【0108】

##### (第4の実施形態)

図13は、第4の実施形態に係る圧縮時のデータの流れを示すデータフロー図である。なお、第4の実施形態によるデータ圧縮伸長システムの構成も、第2の実施形態とほぼ同様なのでここでは図示を省略する。ただし、空間方向コード再配列演算部39および再生画像生成部48における処理内容が第4の実施形態では多少異なっているので、これについては詳しく説明する。

#### 【0109】

図13に示すように、本実施形態においても第3の実施形態と同様に、17フレーム分の画像に対して圧縮を行っているが、空間方向コード再配列演算部39で行う差分演算の内容が上述の例とは異なっている。すなわち、第3の実施形態では、第1フレーム目の画像のコード番号と残りの各フレームのコード番号との差分を夫々演算していたが、第4の実施形態では、互いに隣り合うフレームどうしでコード番号の差分を演算する。

#### 【0110】

すなわち、図13に示すように、まず第1フレーム目と第2フレーム目とで同じアドレスどうしのコード番号間で差分をとり、次に第2フレーム目と第3フレーム目とで同じアドレスどうしのコード番号間で差分をとる。以下、第16フレーム目と第17フレーム目とで差分をとるまで同様の処理を行う。そして、同じアドレスが示すブロックの差分結果をフレーム画像順に並べ替えることにより、時間軸方向に分散して存在する同アドレスブロックの各差分コード番号を1つの

データ列としてまとめて表現する。

#### 【0111】

また、第4の実施形態による伸長側の構成および動作も、第3の実施形態の場合とほぼ同じであるが、再生画像生成部48で行う加算の内容が異なっている。すなわち、第4の実施形態では、互いに隣り合うフレームの空間方向ベクトルを構成しているコード番号どうしで加算を行い、それぞれの加算結果でコード番号のデータ列を再構成する。

#### 【0112】

すなわち、まず、圧縮側より送られてきている第1フレーム目の空間方向ベクトル後のコード番号と、再生画像生成部48により再生された1つ目の空間方向ベクトルを構成しているコード番号差分との加算を行い、第2フレーム目の画像の空間方向ベクトルを生成する。

#### 【0113】

次に、生成された第2フレーム目の画像の空間方向ベクトルを構成しているコード番号と、再生画像生成部48により生成された2つ目の空間方向ベクトルを構成しているコード番号差分との加算を行い、第3フレーム目の画像の空間方向ベクトルを生成する。同じ操作を残りの空間方向ベクトルについても順番に行うことにより、第1フレーム目を除く16フレーム分の空間方向ベクトルのコード番号列を再構成する。

#### 【0114】

なお、この第4の実施形態でも、1つのベクトルを構成するデータの数が16個であり、空間方向用コードブックおよび時間軸用コードブックがそれぞれ2048個のベクトルで構成されているとしたが、これは本実施形態を説明するにあたって分かりやすくするために挙げた例であり、ここで述べた数値に限定されるべきでないのは言うまでもない。

#### 【0115】

また、ここでは17フレーム分の画像をベクトル量子化により圧縮することについて述べたが、圧縮を行うフレームの枚数は17フレームに限定されるものではなく、必要な枚数のフレームをベクトル量子化により圧縮しても良いことは言

うまでもない。また、ここでは動画像を用いた実施形態を示しているが、動画像に限らず、音声信号を対象にしても同様に適用することができる。

#### 【0116】

また、ここではフレーム間のコード番号の差分をとる方法として、隣り合うフレームどうしのコード番号差分をとっているが、必ずしも隣り合うフレーム間のコード番号の差分をとる必要はないことは言うまでもない。

#### 【0117】

以上のように、第2～第4の実施形態によれば、空間方向ベクトル量子化の処理に加えて時間軸ベクトル量子化の処理を導入することにより、動画像の各フレーム内でデータ圧縮を行うだけでなく、各フレーム間に対してもデータ圧縮を行うことができ、より高い圧縮率を得ることができる。

#### 【0118】

すなわち、1枚のフレーム画像に対して空間方向ベクトル量子化のみを行ったときの圧縮率は1／11.6程度であるが、時間軸ベクトル量子化を導入して例えば16フレーム分の動画像に対してデータ圧縮を行うと、圧縮率は16倍（圧縮の対象となるフレーム枚数倍）になり、1／185.6となる。

#### 【0119】

##### (第5の実施形態)

本実施形態では、圧縮した画像を再生する際に、より高画質な再生画像を得るための手法の具体例について述べる。なお、時間軸方向に画像データを圧縮する手法については、第2～第4の実施形態で述べた何れの手法を用いてもよいが、ここでは第2の実施形態の手法を用いた場合を例にとって説明する。

#### 【0120】

上述した第2の実施形態では、空間方向コード再生部44により再生された307200個の空間方向コード番号列を再配列して生成した16個の空間方向ベクトルに対して、それらを構成する個々のコード番号に対応するコードベクトルを空間方向用コードブック記憶部49内の空間方向用コードブックから参照し、該当するパターン画像のデータを各ブロック位置にはめ込むことで元の画像を再生していた。

## 【0121】

ところが、この手法だと1枚目のフレームの再生画像は画質の高いものが得られるが、フレームが進むにつれて時間軸方向のベクトル量子化による量子化誤差が目立つようになり、画質が劣化してくる。そこで、本実施形態では、このような画質の劣化を防ぎ、かつ圧縮率を同等に維持するために、伸長側の再生画像生成部48内の構成を例えば図14のようにする。

## 【0122】

図14において、時間軸コード再配列演算部48aは、第2の実施形態と同様に、空間方向コード再生部44により再生された307200個の空間方向コード番号列を再配列して、19200個のデータ列から成る16個の空間方向ベクトルを生成する。本実施形態では、この時間軸コード再配列演算部48aの後段に時間軸シフト部48bを設けている。

## 【0123】

時間軸シフト部48bは、空間方向デコード部48cが16個の空間方向ベクトル列の個々のコード番号に対応するコードベクトルを空間方向用コードブック記憶部49内のコードブックから参照し、参照してきたパターン画像を各フレーム画像ごとにめ込む際に、圧縮側でベクトル量子化を行ったときに定義したブロックの所望のアドレス毎に、空間方向ベクトルの要素を時間軸方向にずらしてパターン画像をめ込むようにシフト処理を行う。

## 【0124】

すなわち、時間軸シフト部48bは、空間方向ベクトル内の各要素（各アドレス）毎に、近接する要素を時間軸方向にずらす処理を行う。例えば、ある指定されたアドレスについては、16個の空間方向コード番号列が第1フレーム目から順に配列されるが、他のアドレスについては、16個の空間方向コード番号列が第1フレーム目以外のフレームから順に配列されるようにする。

## 【0125】

このようにすることで、時間軸方向のベクトル量子化により生じる量子化誤差を時間軸方向に分散させることができる。よって、フレームが進むごとに目立っていた量子化誤差を目立たなくすることができ、全体的に再生画像の画質を向上

させることができるという利点が得られる。

#### 【0126】

##### (第6の実施形態)

上述した第5の実施形態では、伸長側においてパターン画像を時間軸方向にずらしてはめ込むことにより、量子化誤差を目立たなくさせていた。これに対して、圧縮側において時間軸方向へのシフト処理を行うことにより、量子化誤差を目立たなくさせることもできる。この第6の実施形態では、そのような手法について説明する。

#### 【0127】

本実施形態においても、時間軸方向に画像データを圧縮する手法については、上記第2～第4の実施形態で述べた何れの手法を用いてもよいが、ここでも第2の実施形態の手法を用いた場合を例にとって説明する。

本実施形態において、図9の空間方向コード再配列演算部39は、空間方向ベクトル量子化後のコード番号列を時間軸方向にシフトする処理を行う。

#### 【0128】

すなわち、空間方向コード再配列演算部39は、相関度演算部36による空間方向ベクトル量子化によって各フレーム画像毎に生成された空間方向コード番号列のうち、同じアドレスが示すブロックのコード番号について、アドレス毎に異なるフレームを基準として時間軸ベクトルの要素が配列されるように並べ替えを行う。

#### 【0129】

つまり、上述した第2の実施形態では、空間方向ベクトル量子化後のコード番号列のうち、同じアドレスが示すブロックのコード番号をフレーム画像順に並べ替えて時間軸ベクトルを生成する際に、全てのアドレスについて第1フレームを基準として並べ替えを行っていたが、本実施形態においては、どのフレームを基準として並べ替えるかをアドレス毎に異ならせる。

#### 【0130】

例えば、ある時間軸ベクトルについては、第1フレーム目のコード番号が新しく作られる時間軸ベクトルのデータ列の0番地の位置に置かれるが、他の時間軸

ベクトルについては、第1フレーム目以外のフレームのコード番号が新しく作られるデータ列の0番地の位置に置かれるようにする。

#### 【0131】

このようにすることで、上記第5の実施形態と同様に、時間軸方向のベクトル量子化により生じる量子化誤差を時間軸方向に分散させることができる。したがって、圧縮した画像を再生する際に、フレームが進むごとに目立っていた量子化誤差を目立たなくすることができ、全体的に画質を向上させることができるという利点が得られる。

#### 【0132】

なお、以上の実施形態では空間方向ベクトル量子化と時間軸ベクトル量子化と組み合わせて行う場合を例にとって説明しているが、少なくとも時間軸ベクトル量子化だけを行うようにしても良い。その際に、各フレーム画像内の画素あるいはブロックを単位として、上述のようにフレーム間で（時間軸方向に）位置をずらす処理を行うことにより、時間軸方向のベクトル量子化により生じる量子化誤差を目立たなくすることができる。

#### 【0133】

(第7の実施形態)

図15は、第7の実施形態によるデータ圧縮システムの一構成例を示す機能ブロック図であり、図16は、第7の実施形態に係る圧縮時のデータの流れを示すデータフロー図である。なお、図15において、図9に示した符号と同一の符号を付したものは、同一の機能を有するものであるので、これについての詳細な説明は省略する。また、図16では説明の都合上、原画像を1フレームしか示していないが、実際には複数のフレームが画像入力部31より順に入力される。

#### 【0134】

DC成分検出・除去部51は、原画像に対してベクトル量子化を行う際の各ブロック毎に、ブロックの中からDC成分（各画素の最小輝度値）を検出する。また、上記各ブロック毎に検出したDC成分をブロック内の全ての画素値から夫々減算することにより、入力された原画像の各画素値を各ブロック毎にその最小輝度値からの増分だけで表した画像を生成する。

## 【0135】

相関度演算部36は、演算モード指定部52により空間方向ベクトル量子化を行うモードが指定されているときは、空間方向用コードブック記憶部53にあらかじめ記憶されている、基本パターンだけで構成された空間方向用コードブックを用いて、DC成分除去後の入力画像ブロックに対して空間方向ベクトル量子化の処理を行う。ここで、基本パターンとは、例えば、コードベクトルのブロック内で各画素の輝度値が8つの方向に単調に変化するパターンを言う。

## 【0136】

すなわち、基本パターンのコードブックとは、例えば $4 \times 4$ 画素単位で構成されるブロックのエッジ部分（上下左右の各辺および四隅の各点）の何れかを始点として、ブロック内を上下左右方向あるいは斜め方向に輝度値が徐々に変化するパターンのコードベクトルの集合を言う。これらのコードベクトルの始点の輝度値を0に設定することで、DC成分検出・除去部51により生成されるDC成分の除去された画像ブロックに合ったものとなっている。始点から終点までの輝度値の増分にバリエーションを与えること等により、例えば321パターンのコードベクトルによりこの基本パターンの空間方向用コードブックを構成する。

## 【0137】

また、DC成分検出・除去部51は、各ブロック毎に検出した最小輝度値をそれぞれ抽出する。これにより、図16に示すように、各ブロックの最小輝度値のみで構成した $160 \times 120$ 画素の画像を生成する。このようにして、DC成分検出・除去部51による最小輝度値の抽出処理、および相関度演算部36による空間方向ベクトル量子化処理を行うことで、原画像として入力される複数フレームのそれぞれの画像に対して、空間方向コード番号のデータ列と各ブロックの最小輝度値のデータ列とを生成する。

## 【0138】

次に、空間方向コード再配列演算部39は、上記相関度演算部36によって空間方向用コードブックのコード番号列で表現された各フレームの画像に対して、同じアドレスが示すブロックのコード番号をフレーム画像順に並べ替えることにより、時間軸方向に分散して存在する同アドレスブロックの各コード番号を1つ

のデータ列としてまとめた時間軸ベクトルを生成する。この処理は、上記第2～第4の実施形態で述べた何れの手法によっても良い。

#### 【0139】

同様に、空間方向DC成分再配列演算部54は、上記DC成分検出・除去部51により各ブロックの最小輝度値のみで表現された各フレームの画像に対して、同じアドレスが示すブロックのデータ値をフレーム画像順に並べ替えることにより、時間軸方向に分散して存在する同アドレスブロックの各最小輝度値を1つのデータ列としてまとめた時間軸ベクトルを生成する。この処理も、上記第2～第4の実施形態で述べた何れの手法によっても良い。

#### 【0140】

図16に示した例の場合、基本パターンの時間軸ベクトルについては、例えば第1フレーム目～第16フレーム目の各フレームの先頭アドレスで指定されるブロックのコード番号を並べ替えた(127, 287, 58, …, 283, 276)のデータ列が、1つの基本パターンの時間軸ベクトルに相当する。また、最小輝度値の時間軸ベクトルについては、例えば第1フレーム目～第16フレーム目の各フレームの先頭アドレスで指定されるブロックの最小輝度値を並べ替えた(60, 50, 58, …, 76, 77)のデータ列が、1つの最小輝度値の時間軸ベクトルに相当する。

#### 【0141】

上記空間方向コード再配列演算部39および空間方向DC成分再配列演算部54により並べ替えが行われると、時間軸ベクトル量子化開始信号が出力され、それが演算モード指定部52に与えられる。これに応じて演算モード指定部52は、基本パターンあるいは最小輝度値の何れかについて時間軸ベクトル量子化を行うモードに切り替える。

#### 【0142】

相関度演算部36は、演算モード指定部52により基本パターンの時間軸ベクトル量子化を行うことが指定されているときは、上記空間方向コード再配列演算部39により生成された各時間軸ベクトルに対して、時間軸用コードブック記憶部55に記憶されている時間軸用コードブックを用いてベクトル量子化を行う。

また、演算モード指定部52により最小輝度値の時間軸ベクトル量子化を行うことが指定されているときは、上記空間方向DC成分再配列演算部54により生成された各時間軸ベクトルに対して、時間軸DC成分用コードブック記憶部56に記憶されている時間軸用コードブックを用いてベクトル量子化を行う。

#### 【0143】

以上の処理により、空間方向ベクトル量子化後のコード番号列で表現されていた基本パターンに関する複数の時間軸ベクトルを、基本パターンの時間軸コードブック中の各コードベクトルに対応するコード番号列で置き換えるとともに、各ブロックの最小輝度値のデータ列で表現されていた最小輝度値に関する複数の時間軸ベクトルを、最小輝度値の時間軸コードブック中の各コードベクトルに対応するコード番号列で置き換える。そして、これらのデータリストを伸長側に転送する。

#### 【0144】

なお、本実施形態において、伸長側の構成およびデータの流れについては図示していないが、基本的には圧縮側とは逆の関係になっており、例えば以下のようにして処理を行えば良い。すなわち、基本パターンおよび最小輝度値の各々について、上記第2～第4の実施形態で述べたのと同様の伸長処理を行う。そして、それについて伸長処理が完了したら、それぞれの結果を各ブロック毎に合成していくことによって、元の画像を再現することができる。

#### 【0145】

以上詳しく述べたように、第7の実施形態によれば、原画像から1つの特徴量データ（上述の例では各ブロック毎の最小輝度値）を抽出し、抽出した特徴量データのベクトル量子化と、特徴量除去後の基本パターンのベクトル量子化とを別個に行うようにしたので、様々な特徴が絡み合った原画像そのものに対するコードブックに比べて、それぞれの画像のベクトル量子化で使用するコードブックを単純化することができ、必要なパターン数も減らすことができる（第2～第6の実施形態では2048パターンであったのが、本実施形態では321パターンで済んでいる）。

#### 【0146】

これにより、ベクトル量子化により類似度の大きいパターンをコードブック中から探し出す際に、単純化されたパターンどうしでの比較を行っているために、より適切なパターンに絞り込みやすくなり、高精度のベクトル量子化を行うことができる。よって、このようにして圧縮されたデータを再生する際には、より高画質の再生画像を得ることができるようになる。

#### 【0147】

なお、この第7の実施形態では、抽出する特徴量として最小輝度値を例に挙げて説明したが、これに限定されるものではない。例えば、ブロック内の各画素の最大輝度値や平均値などを抽出するようにしても良い。

また、この第7の実施形態でも、1つの相関度演算部36を空間方向ベクトル量子化と時間軸ベクトル量子化とで使い回すようにしているが、それぞれのベクトル量子化用に複数設けても良い。

#### 【0148】

また、本実施形態は、モノクロ画像やカラー画像を問わずに適用することが可能である。例えば、Y信号（輝度信号）とU、V信号（色信号）とから成るカラー画像の場合、Y信号に対して上述のように最小輝度値、最大輝度値、平均値などを特徴量として抽出することが可能である。また、輝度信号に限らず、色信号に対しても最小値、最大値、平均値等を特徴量として抽出してベクトル量子化を行うようにしても良い。

#### 【0149】

また、以上の例では原画像から1つの特徴量のみを抽出しているが、異なる種類の特徴量（例えば、ブロック内の最小輝度値、輝度変化の方向など）を複数抽出してベクトル量子化を行うようにしても良い。この場合は、出力されるコード番号のリストが増える分圧縮率が低下するが、再生画像の画質は更に向上させることができる。画質を優先させたい場合は、多くの特徴量を抽出してそれぞれ別個にベクトル量子化するようにすれば良い。また、画質と圧縮率のどちらを優先するか、あるいはどの程度優先するかを切り替えられるようにしても良い。

#### 【0150】

（第8の実施形態）

上述したように、動画像に対して圧縮を行う際に、動画を構成する各々のフレーム画像（静止画）に対して単に空間方向ベクトル量子化を行うだけでは、高い圧縮率を得ることは難しい。第2～第7の実施形態では、更に時間軸ベクトル量子化も行うことによって圧縮率を更に高めるようにしてきた。これに対して以下に述べる第8の実施形態では、今までとは異なる手法によって圧縮率を高める手段について説明する。

#### 【0151】

本実施形態によるデータ圧縮システムの構成例を、図17に示す。図17において、画像入力部61は、画像を連続して取り込む装置である。ここでは、例えば1画素が8ビットの階調を持った $640 \times 440$ 画素の画像を1秒間に30枚の割合で取り込む。この画像入力部61は、例えば、画像取り込み用のカメラやビデオカメラのみならず、画像を記憶可能な半導体メモリ装置、ハードディスクやフロッピーディスクなどの記憶媒体によって実現することもできる。

#### 【0152】

コードブック記憶部62には、例えば $4 \times 4$ 画素のブロックで構成されたパターン画像（コードベクトル）が多数登録されていて、各々のブロックにはユニークなコード番号が割り当てられている。

#### 【0153】

コードブック方式による圧縮部63は、画像入力部61で得られた画像の1枚1枚に対して、次に述べるような空間方向ベクトル量子化の処理を行う。すなわち、原画像の左上を起点としてそこから右方向へ向かって順次 $4 \times 4$ 画素のブロックを取り出す。右端まで取り出したら、取り出す位置を1ブロック分下にずらして左端から再び取り出して行き、それを繰り返すことにより全画面分のブロックを取り出す。

#### 【0154】

そして、取り出した各ブロックに対して、コードブック記憶部62に多数登録されているコードベクトルの中から最もパターンの似ているものを選び出し、それに対応するコード番号を出力する。 $640 \times 480$ 画素の画像を処理する場合、19200個のブロックが取り出されて処理されるので、出力されるコード番

号も19200個である。

#### 【0155】

コード列記憶部64は、ある時刻tにおけるフレーム画像をコードブック方式による圧縮部63にて処理して出力されるコード番号列を記憶しておくためのものである。また、コード列比較部65は、コード列記憶部64に記憶されている上記ある時刻tの画像のコード番号列と、次の時刻t+1の画像のコード番号列とを比較し、同じアドレスが示すブロックのコード番号どうしでその差分絶対値をそれぞれ計算する。

#### 【0156】

出力部66は、コード列比較部65から与えられる差分絶対値をもとに、媒体67（ネットワークなどの通信チャンネルまたは記録媒体）に出力するデータを決定する。すなわち、あるアドレスについての差分絶対値がある閾値より大きい場合は、そのアドレスとコード番号とを媒体67に出力する。一方、あるアドレスについての差分絶対値がある閾値より小さい場合は、そのアドレスとコード番号は媒体67に出力しない。

#### 【0157】

次に、上記図17に示したデータ圧縮システムをソフトウェアで実現する場合の構成例を、図18に示す。この図18は、CPU68を含むコードブック方式動画像圧縮プロセッサの構成を示すものである。なお、この図18に示した画像入力部61、コードブック方式による圧縮部63および媒体67は、図17に示したものと同じである。

#### 【0158】

図18において、CPU68に付随して設けられるメモリ69には、コードブック記憶エリア69a、コード列記憶エリア69b、ポインタ・カウンタエリア69cおよび閾値記憶エリア69dが設けられている。

#### 【0159】

コードブック記憶エリア69aは、コードブック方式による圧縮部63で使用するコードブックを記憶しておくものである。コード列記憶エリア69bは、コードブック方式による圧縮部63から各フレーム毎に出力されるコード番号列を

順次記憶しておくものであり、時刻  $t$  の画像をコードブック方式による圧縮部 63 で圧縮した結果のコード番号列が記憶される。

#### 【0160】

また、ポインタ・カウンタエリア 69c は、コード列記憶エリア 69b に対するコード番号の読み出し・書き込みを行う位置を示すポインタ（アドレス）を記憶するものである。閾値記憶エリア 69d は、出力部 66 で媒体 67 に転送するデータを決定するための判断基準となる閾値を記憶するものである。

#### 【0161】

図19は、図18のCPU68が実行する圧縮処理の手順を示すフローチャートである。図19において、まずステップS21で、コードブック記憶エリア 69a からコードブック方式による圧縮部 63 に対してコードブックのデータを転送する。次に、ステップS22で閾値記憶エリア 69b に所定の閾値を設定し、ステップS23でポインタ・カウンタエリア 69c のポインタ（アドレス）をイニシャライズして、ポインタ値を例えば“0”に設定する。

#### 【0162】

次に、ステップS24で、画像入力部 61 から原画像中のマクロブロック（例えば、 $4 \times 4$ 画素のブロック）を1つ読み込む。そして、ステップS25で、読み込んだマクロブロックをコードブック方式による圧縮部 63 に送り、ここで空間方向ベクトル量子化の処理を実行させる。これにより、圧縮コードとして空間方向のコード番号列を得る。

#### 【0163】

次に、ステップS26で、コードブック方式による圧縮部 63 から出力されたコード番号と、以前の処理によって得られコード列記憶エリア 69b に記憶されているコード番号で、現在のポインタ値で示されるコード番号との差分絶対値を計算する。そして、ステップS27で、上記計算された差分絶対値が、閾値記憶エリア 69d に記憶されている閾値よりも大きかどうかを判断する。

#### 【0164】

ここで、差分絶対値の方が大きい場合には、ステップS28で、上記コードブック方式による圧縮部 63 から出力されたコード番号と、そのときのポインタの

値とを媒体67に出力した後、ステップS29で当該コード番号をコード列記憶エリア69bに記憶する。

#### 【0165】

一方、計算された差分絶対値が、閾値記憶エリア69dに記憶されている閾値よりも小さい場合には、ステップS28の処理を実行せずにステップS29に進み、コードブック方式圧縮部63から出力されたコード番号をコード列記憶エリア69bに記憶する。そして、ステップS30でポインタ・カウンタエリア69cのポインタをインクリメントし、ステップS31で1つの画像内の全てのマクロブロックについて処理が終了したかどうかを判断する。

#### 【0166】

全てのマクロブロックについて処理が終了していない場合は、ステップS24に戻り、次のマクロブロックを1つ取り込んで同様の処理を行う。上述した処理は、画像入力部61から得られた画像中のすべてのマクロブロックを処理するまで続けられる。画像中のすべてのマクロブロックを処理し終わると、動画像1フレーム分に対する圧縮処理は全て終了する。以上の処理を各フレーム毎に順次繰り返すことにより、動画像の圧縮が実行される。

#### 【0167】

一方、伸長側においては、送られてきたコード番号に対応するコードベクトルをコードブック中から探し出し、それを各ブロック位置にはめ込んでいくという基本的な処理は、以上に述べた各実施形態と同様である。ただし、本実施形態においては、あるアドレスのブロックについてはコード番号が転送されてきていないので、そのようなブロックについては何らかの方法によって代わりのコードベクトルもしくはパターン画像を用意する必要がある。

#### 【0168】

そのために、例えば、データ伸長システムの構成として、ある時刻tにおけるデコード処理によって再現されたフレーム画像を記憶しておくための再現画像記憶部を設ける。そして、時刻t+1における再現画像をデコード処理によって生成する際に、あるブロックについて圧縮側からコード番号が送られてきていない場合は、そのブロックの画像を上記再現画像記憶部に記憶されている前フレーム

の画像で代用するようとする。このように、コード番号が送られたブロックと送られていないブロックとを伸長側で識別するために、コード番号と共に送られてくるブロックのアドレスが利用される。

#### 【0169】

以上のように、本実施形態では、複数フレームの画像のそれぞれに対して空間方向ベクトル量子化の処理を行い、それぞれのフレーム画像について空間方向コード番号列を生成する。そして、複数フレームの画像の同じアドレスどうしのコード番号について、フレーム間における変化量（差分絶対値）を求め、その変化量がある閾値以上となった場合のみそれを転送データとし、変化量が閾値より小さい場合には転送データとしないようにしている。

#### 【0170】

つまり、複数フレームの画像を構成する各ブロックのうち、フレーム間でデータ値が大きく変化したところのデータだけを転送し、変化が少ないところのデータは転送せずに前フレームのデータを用いるようにする。これにより、動きの激しい部分のデータのみを転送することとなり、個々のフレーム内で空間方向ベクトル量子化した結果に加えて、フレーム方向（時間軸）に対してもデータを圧縮することができ、単なる空間方向ベクトル量子化だけでは得られなかつた高い圧縮率を得ることができる。

#### 【0171】

##### (第9の実施形態)

以上の示した各実施形態は、モノクロ画像に対してもカラー画像に対しても同様に適用することができる。以下に述べる第9の実施形態は、特にカラー画像に対してベクトル量子化によるデータ圧縮を行う場合に、伸長側での再生画像の画質と圧縮率をより向上させることができるようにしたものである。

#### 【0172】

図20は、本実施形態によるデータ圧縮システムの構成例を示すブロック図である。図20において、画像入力部71より入力されたカラーの原画像は、画像メモリ72に一旦蓄えられ、読み出し部73からのアドレス制御等により読み出されて出力される。画像メモリ72より読み出されたカラー画像は、Y, U, V

信号分離部74に与えられる。

#### 【0173】

このY, U, V信号分離部74は、原画像からY信号（輝度信号）とU信号およびV信号（色信号）をそれぞれ分離する。ブロック化部75は、上記Y, U, V信号分離部74により分離された各画像データを、例えば $4 \times 4$ 画素単位のブロックに分割する。このときブロック化部75は、Y, U, Vのどの信号に対してブロック化を行っているかを信号種指定信号として空間方向用コードブック記憶部76に伝える。

#### 【0174】

相関度演算部77は、このようにして分離抽出したY信号、U信号、V信号のそれぞれに対して独立に、上述した類似度を算出する。そして、コード決定部78は、上記相関度演算部77による類似度の演算結果に基づいて、類似度の最も大きいコードベクトルに対応するコード番号をそのブロックの圧縮コード（Winner code）として出力する。

#### 【0175】

このような空間方向ベクトル量子化の実行の際、相関度演算部77では、空間方向用コードブック記憶部76にあらかじめ記憶されているコードブックを用いてベクトル量子化を行うが、空間方向用コードブック記憶部76内のどのコードブックを利用するかは、上記信号種指定信号に応じて決められる。すなわち、空間方向用コードブック記憶部76には、Y信号用のコードブックと、U信号用のコードブックと、V信号用のコードブックとがあらかじめ多数記憶されており、それぞれの信号用のコードブックが、ブロック化部75から相関度演算部77に出力される信号種に合わせて順次利用される。

#### 【0176】

このとき、本実施形態では、Y信号に対してUV信号よりも大きなコードブックを割り当てるようとする。例えば、Y信号用のコードブックを8192個のコードベクトル（13ビット）で構成し、UV信号用のコードブックをそれぞれ128個のコードベクトル（7ビット）で構成するようとする。ただし、Y, U, Vを合わせた全体のコードブックの大きさは、それぞれの信号用のコードブック

を均等に割り当てた場合の全体の大きさと変わらないか、もしくはそれより小さくなるようにする。

#### 【0177】

一般に、画質を決める最も重要な信号はY信号（輝度信号）であるため、それぞれの信号に対して割り当てるコードブック（コードベクトルのコード長）を、UV信号よりもY信号の方を大きくすることにより、全体的なコード長は変えずに再生画像の画質を向上させることができる。

#### 【0178】

すなわち、圧縮レートを $Y : U : V = 1 : 1 : 1$ として圧縮を行った場合、Y信号に対してコードベクトルの数をUV信号よりも多く割り当ても、全体としてのコード長が変わっていないので、1枚の画像に対する圧縮率は変化しない。しかも、Y信号に対してのコードベクトルの数が多いので、画質にとってより重要度の大きいY信号に関して、より類似度の高いコードベクトルをベクトル量子化によって得ることができ、再生画像の画質は向上する。

#### 【0179】

なお、圧縮レートが $Y : U : V = 1 : 1 : 1$ の場合に限らず、 $4 : 1 : 1$ や $4 : 2 : 2$ の場合でも同様の効果を得ることができる。例えば、圧縮レートが $Y : U : V = 4 : 1 : 1$ の場合は、Y信号の各ブロックに対して13ビットのコードを割り当てるのに対して、UV信号に対してはそれぞれ4つのブロックで7ビットのコードを割り当てることになる。しかし、人間の視覚特性として色の誤差に関しては比較的鈍感という性質があるので、このようにしても色再現性に関して特に問題はない。

#### 【0180】

また、以上の実施形態では、Y、U、V各信号の空間方向ベクトル量子化手段として1つの相関度演算部77を使い回すようにしているが、それぞれの信号用に空間方向ベクトル量子化手段を複数設け、各信号を並列に処理するようにしても良い。

また、以上の実施形態では、1つの空間方向用コードブック記憶部76の中にY、U、V各信号用のコードブックを記憶しているが、それぞれのコードブック

を異なる記憶部に分けて記憶するようにしても良い。

### 【0181】

#### (第10の実施形態)

次に、1枚の画像に対して空間方向ベクトル量子化を行う際に用いる空間方向用コードブックの作成方法について説明する。本実施形態では、従来から良く知られている Kohonen の学習アルゴリズムを利用する。図 21 は、この Kohonen の学習アルゴリズムの手順を示す図である。

### 【0182】

この手法では、図 21 に示すように、学習の際に与える初期のコードブックを用意するとともに、学習に用いる 1 つのサンプル画像を用意する。学習の際は、このサンプル画像から例えば  $4 \times 4$  画素単位の画像ブロックを順次入力し、コードブックを用いて空間方向ベクトル量子化の処理を行うことにより、入力した画像ブロックと最も似通ったコードベクトルを検出する。そして、この結果をもとにコードブックの内容を書き換える。

### 【0183】

コードブックの書き換えは、Kohonen の学習アルゴリズムを表す数式中の利得  $k$  および書き換えの範囲  $N_c$  を適当な値に決めて行う。このようなコードベクトルの検出および書き換え処理を所定の学習回数だけ繰り返すことにより、最適化されたコードブックを得る。本実施形態においては、このような学習を実施するに当たって、初期のコードブックとして様々なパターンのものを用いた。

### 【0184】

図 22 は、初期のコードブックとして、とり得る値の最小値から最大値まで輝度値が連続的に変化するパターン (a) と、輝度値がランダムに変化するパターン (b) と、輝度値が 128 (中間値) で一定のパターン (c) とを入力し、これらを用いてコードブックを最適化する。そして、それにより得られたコードブックを用いてベクトル量子化により画像を圧縮・伸長したときの P S N R (peak signal to noise ratio) 特性の違いを示した図である。なお、(a) ~ (c) 中に示す小さい四角の 1 つ 1 つがコードベクトルを表している。この図 22 より、輝度値が連続的に変化するパターンを初期のコードブックとして入力した場合

に、より良好な結果が得られていることが分かる。

#### 【0185】

また、図23は、(a)のようにマップ上（仮想の2次元平面）において1回の学習で書き換えを行う範囲を1次元的に適用し、更新回数の増加とともに書き換え範囲を減少させていく方法と、(b)のように書き換えを行う範囲を2次元的に適用し、更新回数の増加とともに書き換え範囲を減少させていく方法とでコードブックを作成し、更に画像を圧縮・伸長したときのP S N R特性の違いを示した図である。

#### 【0186】

なお、図23(a)および(b)に示す白丸および黒丸の1つ1つがコードベクトルのブロックを示しており、黒丸のブロックは現在のベクトル量子化の対象ブロックを示している。また、これらの白丸および黒丸を囲んだ四角の範囲が、コードブックの書き換え範囲を表している。また、矢印は書き換え範囲を減少させる方向を表している。この図23より、(a)の場合が(b)の場合よりも優れていることが分かる。

#### 【0187】

また、図24は、Kohonenの学習式中の利得kの値を、コードブックの更新回数の増加にかかわらず初期値から減少させない場合と、更新回数の増加とともに初期値から減少させた場合とのP S N R特性の違いを示す図である。この図24より、利得kの値をコードブックの更新回数の増加とともに初期値から減少させていった場合の方が優れていることが分かる。

#### 【0188】

また、図25は、利得kの初期値として様々な値を与え、更新回数を例えば160000回としてコードブックを作成したときのP S N R特性の違いを示す図である。この図25より、利得kの初期値が0.3~1の場合に優れた結果が得られることが分かる。

#### 【0189】

以上の図22~図25に示される結果から、様々なパラメータの中から最も画質を向上させるパラメータの値を特定して学習を行うことにより、それにより生

成したコードブックをデータ圧縮伸長システムに用いることで、再生画像の画質を向上させることができる。例えば、学習の際に与える初期のコードブックとして、とり得る値の最小値から最大値まで輝度値が連続的に変化するパターンを用いる。また、1回の学習で影響を与える範囲は1次元的に適用し、かつ更新回数の増加と共にその範囲を減少させていく。さらに、学習の際に与える利得係数kの初期値を0.3～1の範囲の何れかの値とし、かつ更新回数の増加と共に係数値を減少させていくことによって、高精度なコードブックを得ることができ、これを用いて圧縮・伸長した再生画像の画質を向上させることができる。

#### 【0190】

##### (第11の実施形態)

本実施形態は、学習により複数のコードブックを作成し、それらを合成して新しいコードブックを作成することにより、高精度なコードブックを得るようにしたものである。図26は、最適化で得られた3種類のコードブックA, B, Cの特徴を1つのコードブックA+B+Cにまとめる手法を示している。

#### 【0191】

図26において、コードブックA+B+Cのサイズ(コードベクトルの個数)は、A, B, C単独のコードブックサイズと同様に、例えば1024である。つまり、1つにまとめたコードブックA+B+Cの中に含まれるパターンは、個々のコードブックA, B, Cに含まれるパターンをそれぞれ適当に間引いて合成したものである。

#### 【0192】

ここでは、Kohonenの自己組織化マップで得られたコードブックは、似たようなパターンがマップ上に近接して存在するため、この特性を生かして各コードブックA, B, Cのマップ上の左上から順番に1つおきにそのままパターンを取り出し、これを新しいコードブックA+B+Cとして再構成している。これにより、それぞれのコードブックA, B, Cの特徴を残したコードブックA+B+Cが新たに生成される。

#### 【0193】

図27は、図26に示したA, B, C, A+B+Cそれぞれのコードブックを

用いて画像A'，B'，C'に対してベクトル量子化を行った後、それを復元した画像のP S N R特性を示す図である。なお、画像A'，B'，C'は、それぞれ単独のコードブックA，B，Cを作成する際に用いたサンプル画像であるものとする。

#### 【0194】

図27から分かるように、単独のコードブックA，B，Cは、それぞれを作成する際に用いたサンプル画像A'，B'，C'に対しては概ね良好な結果を示すが、他の種類の画像に対しては良好な結果は得られていない。例えば、コードブックAを用いた場合、サンプル画像A'に対しては良好な結果が得られるが、他の画像B'，C'に対しては良好な結果は得られない。

#### 【0195】

これに対して、合成したコードブックA+B+Cを用いた場合のP S N R特性は、全ての画像A'，B'，C'について、単独のコードブックA，B，Cを用いた場合の特性と匹敵しており、かつ全ての特性を兼ね備えたものとなっている。このように、特徴的な画像をいくつか組み合わせてコードブックを作成することで、様々な画像に対して耐性を有するコードブックを作成することができる。例えば、人の顔、風景、文字などの画像を用いてそれぞれのコードブックを作成し、それらを合成することにより、コードブックメモリのサイズを大きくすることなく汎用的なコードブックを作成することができる。

#### 【0196】

なお、上記実施形態では、同じサイズの3種類のコードブックを1つにまとめる手法を示したが、3種類に限定されるものではなく、2種類あるいは3種類以上のコードブックをまとめて1つに合成することもできる。また、サイズの異なる複数のコードブックをまとめて1つに合成するようにすることもできる。

#### 【0197】

(第12の実施形態)

ベクトル量子化によりコードブックの中から最も似通ったコードベクトルを選び出すためには、膨大な演算量が必要である。そのため、ベクトル量子化の専用ハードウェアを設計する場合、高速に演算を行うためには通常は膨大なハードウ

エア面積を必要とすることになる。以下に述べる実施形態は、ハードウェア規模を大きくすることなく演算を高速に行えるようにしたものである。

#### 【0198】

図28は、2つの入力ベクトルが類似しているか否かを数値化した類似度を求める関数としてマンハッタン距離を利用し、また、画像の特徴量としてベクトルデータの要素の総和を利用した場合のベクトル量子化装置の構成と、演算が省略される様子とを説明するための図である。なお、類似度としてマンハッタン距離を用いた場合、距離が小さいほうが類似度は大きい。

#### 【0199】

最初に、演算が省略できる原理を説明する。

ベクトルデータの次元数（要素数）をn、コードブック記憶部81内のテンプレートベクトル（コードベクトル）データの総数をkとする。このとき、

入力ベクトルデータ： $I = (I_1, I_2, \dots, I_n)$

テンプレートベクトルデータ： $T_m = (T_{m1}, T_{m2}, \dots, T_{mn})$   
( $m = 1, 2, \dots, k$ )

について、入力ベクトルデータIとテンプレートベクトルデータ $T_m$ との類似度を、以下の式(1)で示すマンハッタン距離 $M_m$ で定義すると、入力ベクトルデータIに最も類似しているテンプレートベクトルデータ $T_m$ を検索する操作とは、全てのマンハッタン距離 $M_m$  ( $m = 1, 2, \dots, k$ )の中で距離 $M_m$ が最も小さくなるmの値を探し出す操作に相当する。

#### 【0200】

##### 【数1】

$$M_m = \sum_{i=1}^n |I_i - T_{mi}| \quad \dots \dots (1)$$

#### 【0201】

ここで、以下の式(2)に示すような、入力ベクトルデータIの要素の総和とテンプレートベクトルデータ $T_m$ の要素の総和との差の絶対値 $D_m$ と、上記式(1)

に示したマンハッタン距離 $M_m$ との間には、 $D_m \leq M_m$ という関係が成り立つことに注目すると、以下のような議論が成り立つ。

## 【0202】

## 【数2】

$$D_m = \left| \sum_{i=1}^n I_i - \sum_{i=1}^n T_{mi} \right| \quad \cdots\cdots(2)$$

## 【0203】

すなわち、コードブック記憶部81内にあるテンプレートベクトルデータ $T_m$ と入力ベクトルデータ $I$ とのマンハッタン距離 $M_m$ が判明しているとする。このとき、コードブック記憶部81内にあって、上記テンプレートベクトルデータ $T_m$ とは別な任意のテンプレートベクトルデータ $T_o$ と入力ベクトルデータ $I$ について、各々のベクトルデータの要素の総和をとり、その総和どうしの差の絶対値 $D_o$ を計算した結果、 $D_o > M_m$ となっている場合を仮定する。

## 【0204】

この場合、上述の $D_m \leq M_m$ という関係により、常に $M_o > M_m$ が成り立つ。よって、テンプレートベクトルデータ $T_o$ と入力ベクトルデータ $I$ とのマンハッタン距離 $M_o$ を計算せども、差分絶対値 $D_o$ を計算するだけで、テンプレートベクトルデータ $T_o$ はテンプレートベクトルデータ $T_m$ に比べて入力ベクトルデータ $I$ に類似していないことが分かる。つまり、テンプレートベクトルデータ $T_o$ を検索対象から除外できることになる。

## 【0205】

以上だけを見ると、テンプレートベクトルデータ $T_o$ について、マンハッタン距離 $M_o$ を計算しなくても、差分絶対値 $D_o$ を計算する必要があるのならば、検索に関わる処理は減少していないように見える。しかし、差分絶対値 $D_o$ の演算において、テンプレートベクトルデータ $T_m$ の特徴量である要素の総和 $\sum T_{mi}$ (式(2)中の右辺第2項)は、あらかじめ計算して特徴量記憶部82に記憶しておくことができる。

## 【0206】

よって、差分絶対値 $D_0$ の算出は、入力ベクトルデータIの特徴量の演算と、これと特徴量記憶部82内に記憶されている特徴量との差をとる演算を行うだけで済む。また、入力ベクトルデータIの特徴量の算出は、1つの入力ベクトルデータに対して1回の演算で済む。このため、差分絶対値 $D_0$ の算出は、マンハッタン距離 $M_0$ の算出に比べて非常に少ない演算量で行うことができる。

## 【0207】

なお、特徴量としてベクトルデータの要素の総和を用いることと、要素の平均値を用いることとは等価である。なぜならば、要素の平均値とは要素の総和を要素数で割っただけのものに過ぎないからである。

## 【0208】

次に、図28を使って具体的に説明する。

図28において、入力ベクトルデータIおよびコードブック記憶部81内の複数のテンプレートベクトルデータ $T_1, T_2, T_3, T_4, \dots$ は、例として共に6次元のベクトルデータとなっているが、任意の次元でかまわない。特徴量記憶部82には、コードブック記憶部81内に記憶されている各テンプレートベクトルデータ $T_1, T_2, T_3, T_4, \dots$ の特徴量（例えば要素の総和） $\Sigma T_1, \Sigma T_2, \Sigma T_3, \Sigma T_4, \dots$ が記憶されている。

## 【0209】

図28中では、コードブック記憶部81内の各テンプレートベクトルデータ $T_1, T_2, T_3, T_4, \dots$ と、すぐ左側に位置する特徴量記憶部82内の各特徴量 $\Sigma T_1, \Sigma T_2, \Sigma T_3, \Sigma T_4, \dots$ とは対応している。つまり、テンプレートベクトルデータ $T_1$ の特徴量は $\Sigma T_1$ 、テンプレートベクトルデータ $T_2$ の特徴量は $\Sigma T_2$ 、以下同様となる。これらのコードブック記憶部81および特徴量記憶部82内のデータは、アドレスカウンタ88からのアドレスに従って順に読み出される。

## 【0210】

また、マンハッタン距離演算部83は、コードブック記憶部81から転送された $d_2$ で示される各テンプレートベクトルデータ $T_m$ と、 $d_1$ で示される入力ベ

クトルデータ I とのマンハッタン距離  $M_m$  ( $m = 1 \sim k$ ) を求め、d 3 で示されるマンハッタン距離データを出力する。また、最小マンハッタン距離記憶部 8 4 は、演算省略判定部 8 7 による更新制御に従って、今までに計算されたマンハッタン距離  $M_m$  のうち、最小の距離  $m i n M$  を記憶する。

## 【0211】

ある 1 つの入力ベクトルデータ I について全てのテンプレートベクトルデータ  $T_m$  との演算が終了したとき、この最小マンハッタン距離記憶部 8 4 に記憶されているものに対応するテンプレートベクトルデータが、入力ベクトルデータ I に最も類似するデータである。そして、このように最も類似する（最も距離が短い）テンプレートベクトルデータのアドレス  $m i n A$  は、演算省略判定部 8 7 による更新制御に従って、アドレスカウンタ 8 8 から最小距離アドレス記憶部 8 9 に与えられて記憶されている。

## 【0212】

そこで、ある 1 つの入力ベクトルデータ I について処理が終わったときに、この最小距離アドレス記憶部 8 9 に記憶されているアドレスが、当該ブロックのコード番号に相当することになる。なお、このコード番号は、1 つの入力ベクトルデータ I の処理が終わる毎に出力しても良いし、1 枚の画像全体の処理が終わったときにまとめて出力するようにしても良い。

## 【0213】

特微量演算部 8 5 は、入力ベクトルデータ I の各要素の総和を演算し、その結果を出力する。少なくとも同じ入力ベクトルデータ I について類似度を求める演算を行っている間は、ここで演算した特微量は保存されている。特微量差分演算部 8 6 は、上記特微量演算部 8 5 で求められた入力ベクトルデータ I の特微量 d 4 と、特微量記憶部 8 2 から読み出されたテンプレートベクトルデータ  $T_m$  の特微量 d 5 との差分を演算し、出力する。

## 【0214】

演算省略判定部 8 7 は、最小マンハッタン距離記憶部 8 4 より出力される、今までに分かっている最小のマンハッタン距離 d 6 と、特微量差分演算部 8 6 で演算された特微量の差分 d 7 を利用して、テンプレートベクトルデータ  $T_m$  と入

力ベクトルデータIのマンハッタン距離を演算する必要があるか否かを判別する。

#### 【0215】

そして、その判別結果をマンハッタン距離演算部83に伝え、必要でない演算を省略するようにするとともに、最小マンハッタン距離記憶部84、アドレスカウンタ88および最小距離アドレス記憶部89に所定の制御信号を出力し、それぞれの内容の更新制御を行う。

#### 【0216】

アドレスカウンタ88の内容は、1回の処理が終わることに更新されるが、最小マンハッタン距離記憶部84および最小距離アドレス記憶部89の内容は、所定の条件を満たしたときのみ更新される。なお、テンプレートベクトルデータを読み出す前にアドレスカウンタ208の値をインクリメントすれば、マンハッタン距離演算部202への制御信号は必ずしも必要ではない。

#### 【0217】

次に、上記のように構成したベクトル量子化装置のベクトル量子化の手順を示す。演算の省略の様子も同時に説明する。

1) まず、入力ベクトルデータIの特徴量を特徴量演算部85により演算し、その結果を保存する。この特徴量は、1つの入力ベクトルデータIに対して1度だけ演算すればよい。

#### 【0218】

2) 次に、入力ベクトルデータIと、コードブック記憶部81内の第1番目のテンプレートベクトルデータ $T_1$ とのマンハッタン距離 $M_1$ をマンハッタン距離演算部83によって演算する。この演算結果を以下の式(3)に示す。この段階ではマンハッタン距離はまだ1つしか演算していないので、現在までに見つかっている最小のマンハッタン距離 $m_i n M$ は、 $M_1$ となる。したがって、このマンハッタン距離 $M_1$ を最小マンハッタン距離記憶部84に記憶する。

#### 【0219】

## 【数3】

$$M_1 = \sum_{i=1}^6 |I_i - T_{1i}| = 70 \quad \cdots\cdots(3)$$

## 【0220】

3) 次に、特微量記憶部82内の第2番目のテンプレートベクトルデータ $T_2$ の特微量 $\Sigma T_2$ と、特微量演算部85に保存されている入力ベクトルデータ $I$ の特微量（この例では265）との差の絶対値 $D_2$ を特微量差分演算部86で求め る。この演算結果を以下の式(4)に示す。

## 【0221】

## 【数4】

$$D_2 = \left| \sum_{i=1}^6 I_i - \sum_{i=1}^6 T_{2i} \right| = 4 \quad \cdots\cdots(4)$$

## 【0222】

4) さらに、手順3)で求めた特微量の差の絶対値 $D_2$ と、今までに見つかっている最小のマンハッタン距離 $minM$ とをもとに、第2番目のテンプレートベクトルデータ $T_2$ と入力ベクトルデータ $I$ とのマンハッタン距離 $M_2$ を演算する必要があるかどうかを、演算省略判定部87で判断する。今の場合、 $D_2 < minM (=M_1)$ であるから、第2番目のテンプレートベクトルデータ $T_2$ と入力ベクトルデータ $I$ とのマンハッタン距離 $M_2$ の演算は省略できない。

## 【0223】

5) 手順4)により、第2番目のテンプレートベクトルデータ $T_2$ と入力ベクトルデータ $I$ とのマンハッタン距離 $M_2$ の演算が必要と判定されたので、マンハッタン距離演算部83はこの演算を行う。この演算の結果は、 $M_2 = 22$ であるが、これは今までに見つかっている最小のマンハッタン距離 $minM (=M_1)$ よりも小さいので、 $minM$ の値を $M_2$ に更新する。よって、 $minM = M_2$ と

なる。つまり、最小マンハッタン距離記憶部84には第2番目のテンプレートベクトルデータ $T_2$ に関するマンハッタン距離 $M_2$ が記憶される。

## 【0224】

6) 次に、特徴量記憶部82内の第3番目のテンプレートベクトルデータ $T_3$ の特徴量 $\Sigma T_3$ と、特徴量演算部85に保存されている入力ベクトルデータIの特徴量との差の絶対値 $D_3$ を特徴量差分演算部86で求める。この演算結果は、 $D_3 = 46$ である。

## 【0225】

7) さらに、手順6)で求めた特徴量の差の絶対値 $D_3$ と、現在までに見つかっている最小のマンハッタン距離 $minM$ とをもとに、第3番目のテンプレートベクトルデータ $T_3$ と入力ベクトルデータIとのマンハッタン距離 $M_3$ を演算する必要があるかどうかを、演算省略判定部87で判断する。今の場合、 $D_3 > minM (=M_2)$ であるから、第3番目のテンプレートベクトルデータ $T_3$ と入力ベクトルデータIとのマンハッタン距離 $M_3$ の演算は省略できる。

## 【0226】

8) 手順7)により、第3番目のテンプレートベクトルデータ $T_3$ と入力ベクトルデータIとのマンハッタン距離 $M_3$ の演算は省略できると判明したので、この演算は省略する。そして、第4番目のテンプレートベクトルデータ $T_4$ の特徴量 $\Sigma T_4$ と入力ベクトルデータIの特徴量との差の絶対値 $D_4$ を特徴量差分演算部86で演算する。その演算結果は、 $D_4 = 157$ である。

## 【0227】

9) 手順8)で求めた特徴量の差の絶対値 $D_4$ と、現在までに見つかっている最小のマンハッタン距離 $minM$ とをもとに、第4番目のテンプレートベクトルデータ $T_4$ と入力ベクトルデータIとのマンハッタン距離 $M_4$ を演算する必要があるかどうかを、演算省略判定部87で判断する。今の場合、 $D_4 > minM (=M_2)$ であるから、第4番目のテンプレートベクトルデータ $T_4$ と入力ベクトルデータIとのマンハッタン距離 $M_4$ の演算も省略できる。

## 【0228】

10) 以下、同様の処理を $m = k$ となるまで繰り返す。

## 【0229】

図29は、上記した本実施形態のベクトル量子化処理の手順を示すフローチャートである。以下、このフローチャートについて説明する。

図29において、まず最初にステップS41で入力ベクトルデータIを入力する。そして、ステップS42で、その入力ベクトルデータIの要素の総和（特徴量）を演算し、その結果を保存する。

## 【0230】

次に、ステップS43で、検索するテンプレートベクトルデータの番号（アドレス）を $m = 1$ 、 $m \leq n_A = 1$ に初期化するとともに、今までに見つかっている最小のマンハッタン距離 $m_i \leq M$ の値を $\infty$ に初期化する。次に、ステップS44で、アドレスカウンタ $m$ の値がテンプレートベクトルデータの数 $k$ を越えたかどうかを判断し、越えていない場合はステップS45に進む。

## 【0231】

ステップS45では、検索するテンプレートベクトルデータ $T_m$ についてあらかじめ記憶されている特徴量と、演算の結果保存されている入力ベクトルデータIの特徴量との差の絶対値 $D_m$ を求める。そして、ステップS46で、その演算した差分絶対値 $D_m$ が、今までに見つかっている最小のマンハッタン距離 $m_i \leq M$ の値以上かどうかを判断する。

## 【0232】

ここで、差分絶対値 $D_m$ が現在の最小マンハッタン距離 $m_i \leq M$ の値以上の場合は、ステップS50でアドレスカウンタ $m$ の値をインクリメントした後、ステップS44の処理に戻る。一方、上記差分絶対値 $D_m$ が現在の最小マンハッタン距離 $m_i \leq M$ より小さい場合は、ステップS47に進み、入力ベクトルデータIと、検索するテンプレートベクトルデータ $T_m$ とのマンハッタン距離 $M_m$ を演算する。

## 【0233】

そして、ステップS48で、演算したマンハッタン距離 $M_m$ が現在の最小マンハッタン距離 $m_i \leq M$ 以上であるかどうかを判断し、そうである場合はステップS50でアドレスカウンタ $m$ の値をインクリメントした後、ステップS44の処

理に戻る。一方、演算したマンハッタン距離 $M_m$  が現在の最小マンハッタン距離 $\min M$ より小さい場合は、ステップS49に進む。

#### 【0234】

ステップS49では、最小マンハッタン距離 $\min M$ の値を上記演算したマンハッタン距離 $M_m$ に更新するとともに、探し出したテンプレートベクトルデータの番号（アドレス）を最小距離アドレス記憶部89に記録する。その後、ステップS50でアドレスカウンタmの値をインクリメントした後、ステップS44の処理に戻る。

#### 【0235】

以上の手順の中でも示した通り、図28に記載した構成のベクトル量子化装置を用いれば、入力ベクトルデータIに最も類似したテンプレートベクトルデータを検索する操作を行う際に、その検索に必要な類似度の演算を少なくすることができ、高速に演算を行うことができる。

#### 【0236】

なお、図28では、コードブック記憶部81内の各テンプレートベクトルデータ $T_1, T_2, T_3, T_4, \dots$ は任意の順で並んでいたが、特徴量の大きい順に並べても良い。テンプレートベクトルデータが特徴量順に並んでいる場合、テンプレートベクトルデータを検索する際、入力ベクトルデータIと特徴量が最も近いテンプレートベクトルデータから検索をすることが容易になる。特徴量が近い2つのベクトルデータは、類似度も大きい傾向があるので、入力ベクトルデータと特徴量が最も近いテンプレートベクトルデータから検索を行うことで、類似度の演算をより多く省略することができる。

#### 【0237】

また、本実施形態では、ベクトル量子化装置が取り扱うベクトルデータの次元を6次元としたが、ベクトルデータの次元は任意で良い。

また、図28のベクトル量子化装置は、専用のハードウェアで実現することも可能であるし、コンピュータ上のプログラムによってソフトウェアで実現することもできる。ハードウェアで実現する場合、新たに設けた特徴量演算部85、特徴量差分演算部86および演算省略判定部87は、それ程大きなハードウェア面

積を必要としない。

#### 【0238】

また、上記実施形態において、ベクトルデータの特徴量としては、ベクトルデータの要素の総和に限定されるものではない。例えば、ベクトルデータの分散を使いたい場合は、特徴量演算部85と演算省略判定部87とを分散演算用のものに変更すれば良い。また、類似度を求める関数をマンハッタン距離以外の関数、例えばユークリッド距離を求める関数等に変更したい場合は、マンハッタン距離演算部83と最小マンハッタン距離記憶部84と演算省略判定部87とを変更すれば良い。

#### 【0239】

##### (第13の実施形態)

第13の実施形態では、上記第12の実施形態で示したベクトル量子化装置への入力として画像データを用い、上記実施形態のベクトル量子化装置を画像圧縮装置に応用した例を、図30および図31を使って説明する。まず、ベクトル量子化装置を使って画像圧縮を行う場合の一般的な例を図30により説明する。

#### 【0240】

図30において、入力される原画像91は、画素と呼ばれる要素が多数集まって構成されている。個々の画素は輝度値、あるいは色差信号などの情報を持っている。入力画像91中から複数画素で構成されるブロックを取り出したのが、入力画像ブロック92である。図30の例では、入力画像ブロック92の大きさとして $4 \times 4$ 画素を選んでいるが、この大きさは何であっても良い。

#### 【0241】

入力画像ブロック92は、上述の通り複数の画素を持っているから、各々の画素が持つ輝度値などを集めてベクトルデータとすることができる。これが図28で説明したベクトル量子化装置の入力ベクトルデータIである。

#### 【0242】

人間の視覚特性上、入力画像91中の幾つかの入力画像ブロックは、見た目では殆ど同じに見える場合がある。こういった同じに見える複数の入力画像ブロックを、より少ない数の画像ブロックで代表させることが可能である。画像プロッ

クコードブック93は、入力画像91上の多数の入力画像ブロックを代表する画像ブロック（テンプレートベクトルデータ）を複数持つものである。テンプレートベクトルデータは、画像ブロックコードブック93内の画像ブロック各々の画素が持つ輝度をベクトルデータとしたものである。

#### 【0243】

ベクトル量子化装置では、入力画像91全体を画像ブロックとして分割し、各々の画像ブロック92を入力ベクトルデータとして、その入力ベクトルデータに類似するテンプレートベクトルデータをコードブック93内から検索する。そして、該当するテンプレートベクトルデータの番号のみを転送することで、画像を圧縮することが可能である。圧縮された画像を再生して再現画像94を得るには、上記転送された番号に対応するテンプレートベクトルデータをコードブック93から読み出し、画像に当てはめれば良い。

#### 【0244】

本実施形態において、入力画像91中から画像ブロック92で示される入力ベクトルデータを取り出てしまえば、類似度を求める関数としてマンハッタン距離を用い、特徴量としてベクトルデータの要素の総和を用いた場合、ベクトル量子化の手順は上述した第12の実施形態と全く同一である（ここではその詳細な手順の説明は省略する）。よって、入力ベクトルデータに類似するテンプレートベクトルデータを検索する際、検索にまつわる演算を少なくできるので、高速な画像の圧縮が可能となる。

#### 【0245】

以上のような方法によって画像データに対してベクトル量子化の操作を行ったときに、マンハッタン距離の演算をどれほど省略できるのかを示したのが、図31である。図31は、屋外風景写真、屋内風景写真、人物写真を入力画像としたときに、様々なテンプレートベクトルデータ群に対して、各々のテンプレートベクトルデータ群の中でマンハッタン距離を演算しなければならないテンプレートベクトルデータの割合を表している。

#### 【0246】

この図31によると、テンプレートベクトルデータ群の中で、入力ベクトルデ

ータとのマンハッタン距離を演算しなければならないテンプレートベクトルデータは、せいぜい14%以内となっていることが分かり、本実施形態の有用性が示されている。

#### 【0247】

なお、以上の例では画素の輝度値をベクトルデータとしたが、輝度値だけでなく、各画素が有する数値化された情報は何であれベクトルデータとすることができる。また、入力画像91中から入力画像ブロック92を取り出した後、離散余弦変換をはじめとする様々な変換をかけ、変換後の画像からベクトルデータを作ってベクトル量子化装置の入力としても良い。

#### 【0248】

また、入力ベクトルデータが画像データの場合は、特微量としてベクトルデータの要素の総和を用いても、画像データを周波数分解した際の直流成分を用いても、両者は等価である。何故ならば、上記直流成分は、ベクトルデータの要素の総和に対するある係数を掛けたものに過ぎないからである。

#### 【0249】

##### (第14の実施形態)

上記第12の実施形態で示したベクトル量子化装置への入力として画像データを用いた場合、ベクトルデータの特微量として、画像特有の性質を使うことが可能である。本実施形態では、ベクトル量子化装置の入力を画像データとした場合の画像データ特有な特微量について説明し、幾つかの特微量を組み合わせてベクトル量子化処理の演算量を減少させることが可能であることを以下に示す。

#### 【0250】

まず、画像データに対して、類似度を演算する関数をマンハッタン距離とし、特微量としてベクトル要素の総和を用いた場合の問題点を、図32を使って説明する。図32において、入力ベクトルデータに対応する画像ブロック101と、テンプレートベクトルデータに対応する画像ブロック102は、ほぼ同じブロックである。一方、入力ベクトルデータに対応する画像ブロック101と、もう一方のテンプレートベクトルデータに対応する画像ブロック103は、全く上下左右が反転したブロックとなっており、見た目では全く異なるものである。

## 【0251】

実際に、入力画像ブロック101とテンプレート画像ブロック102, 103との画素の輝度値を取り出して、それぞれをベクトルデータとしてマンハッタン距離を演算すると、テンプレート画像ブロック103は、もう一方のテンプレート画像ブロック102と比較して入力画像ブロック101と類似していないことが明らかになる。

## 【0252】

しかしながら、特徴量であるベクトルデータの要素の総和は、2つのテンプレート画像ブロック102と103とで等しい。このことは、特徴量としてベクトルデータの要素の総和を使っただけでは、無駄な類似度演算が多く残ることを意味している。

## 【0253】

この問題を解決するための手段としては、特徴量として、ベクトルデータの各要素の一部を画素の明暗が反転するように操作した後の各要素の総和を使うという手段が考えられる。

## 【0254】

図33は、ベクトルデータの要素の一部の明暗を反転させた場合、画像ブロックがどのように変化するかを表している。初期の画像ブロック111に対して、3種類の反転パターン112～114を用いて、黒で塗りつぶされた部分の画素が明暗反転するように操作した結果が、反転後画像ブロック115～117である。このように、1つの画像に対して異なる反転パターン112～114で明暗反転をすると、初期の画像ブロックが同じでも、反転後画像ブロック115～117の画素の輝度値の総和118～120は異なった値となる。

## 【0255】

また、図34は、ベクトルデータの要素の総和が等しい場合でも、ベクトルデータの要素の一部を画素の明暗が反転するように操作してから、ベクトルデータの要素の総和を求めて特徴量とすると、特徴量に差が現れてくる例を示したものである。図34において、初期の画像ブロック121と122は、画素の輝度値の総和は全く同じである。よって、ベクトルデータの要素の総和、すなわち、ブ

ロック内の画素の輝度値の総和を特徴量としても両者は全く区別できない。

#### 【0256】

このような2つの初期画像ブロック121, 122に対して、同じ反転パターン123の黒塗りの部分を明暗反転し、反転後の画像ブロック124, 125について画素の輝度値の総和126, 127を計算すると、両者は全く異なった値となる。

#### 【0257】

本実施形態は、以上のことを利用して、無駄なマンハッタン距離の演算を更に減少させることができるようにしたのである。すなわち、以上の例からも分かるように、ある特徴量を使ったときには省略できない類似度の演算が、別の特徴量を使うと省略できる場合がある。この場合、異なる特徴量を2つ以上利用して演算量を更に減少させることができる。

#### 【0258】

具体的には、ある特徴量でテンプレートベクトルデータの中から類似度を演算しなければならない範囲を絞り込み、別の特徴量で更に範囲を絞り込むという、2段階で特徴量を使う方法や、同時に2つの特徴量を使って、特徴量の差が大きい方を演算省略の判断に使う方法などが挙げられる。

#### 【0259】

なお、ある画像から入力画像ブロックを取り出してきてベクトル量化装置の入力とする場合、上述の「ベクトルデータの要素の一部を画素の明暗が反転するよう操作した後のベクトルデータの総和」の他にも、画像特有の特徴量がいくつか考えられる。以下では、画像ブロックの四隅の画素の情報を特徴量として使う方法と、画像ブロック上の画素値の変化の様子を特徴量として使う方法とを説明する。

#### 【0260】

最初に、画像ブロックの四隅の画素の情報を特徴量として使う方法について説明する。画像ブロックを入力画像中から取り出してきた場合に、ブロック内で画素値がある方向に滑らかに変化する画像に対しては、画像ブロックの四隅から特徴量を得ることができる。図35は、画像ブロックの四隅を特徴量として使う方

法を説明した図である。

#### 【0261】

図35(a)において、画像ブロック131は、ブロックの右下から左上方向に輝度が徐々に明るくなる変化をしている。この画像ブロック131から四隅133を取り出し、それらの4画素の輝度値を比較すると、画像ブロック131はどのような方向に輝度が変化しているのかを把握することができる。

#### 【0262】

また、画像ブロック132とその四隅134は、画素値の変化する方向が上記とは異なる場合の一例を示すものであり、画像ブロックの右辺から左辺方向に輝度が徐々に明るくなる変化を示している。なお、133および134中に示した矢印は、画素の輝度の大小関係を示すものであり、矢印の先にある画素の方が輝度が大きい。また、輝度の大小に差がない場合には、線分で表している。

#### 【0263】

このように、画像ブロックの四隅の画素の間で輝度を大小比較し、大小を判別することで、画像ブロックの特徴を捉えることができる。四隅の画素の間での輝度の大小パターンは幾通りもあるので、それらのパターンを番号付けして特徴量とすることができます。すなわち、図35(b)に示すように、四隅の画素の間での輝度の大小の各パターンに対して番号を付与し、その番号を特徴量として本実施形態のベクトル量子化装置に組み込むことができる。

#### 【0264】

なお、図35では、画像ブロックの大きさを $4 \times 4$ 画素としているが、この大きさは任意で良い。また、画素値として輝度値を用いているが、他の画素値（例えば色信号値）を用いても良い。

#### 【0265】

次に、画像ブロック上の画素値の変化を特徴量として使う方法を説明する。入力画像中から画像ブロックを幾つか切り出してくると、それらの画像ブロックの中には、画素値の変化の様子が似ているものがある。図36は、画素値の変化の周期および変化率が同じである2つの画像ブロック141、142を示している。これらの画像ブロック141と142は何れも左右方向にのみ画素値が変化

している。

#### 【0266】

画像ブロック141の左右方向の画素値の変化は、もう一方の画像ブロック142の左右方向の画素値の変化に所定の係数をかけて、変化の振幅を抑えたものに他ならない。このような状態を、画像ブロック141, 142の画像ブロック上での画素値の変化のモードが同じであると言うことにする。

#### 【0267】

画像ブロック上での画素値の変化のモードは多数あるので、それらの変化のモードを番号付けすることにより特徴量とすることができます。すなわち、画素値の変化のモードに対してそれぞれ番号を付与し、その番号を特徴量として本実施形態のベクトル量子化装置に組み込むことができる。

なお、この図36でも画像ブロックの大きさを $4 \times 4$ 画素としているが、この大きさは任意で良い。

#### 【0268】

最後に、第1の特徴量として画像ブロック上での画素値の変化、つまり画像ブロック上での変化のモードを番号付けしたものを用いるとともに、第2の特徴量として画像ブロック上の画素の値の総和、つまりベクトルデータの要素の総和を用いた場合の、第14の実施形態に係るベクトル量子化装置の構成を、図37を用いて説明する。なお、図37において、図28に示したブロックと同じブロックには同一の符号を付している。また、ここでは類似度をマンハッタン距離によって求めている。また、図37中ではベクトルデータを、相当する画像ブロックとして視覚的に表現している。

#### 【0269】

入力ベクトルデータIは、第1の特徴量である画像ブロック上での画素値の変化モードによって分類される。すなわち、変化モード判定部151は、画素値の変化モードの典型的なベクトルデータを集めた特徴テンプレート記憶部152に記憶されているベクトルデータの情報をもとに、入力ベクトルデータIの第1の特徴量である画像ブロック上での画素値の変化モードが、テンプレートの何番に相当するものであるかを示すデータd8を出力する。

## 【0270】

テンプレートベクトルデータを記憶するコードブック記憶部81と、テンプレートベクトルデータの第2の特徴量であるベクトルデータの要素の総和を記憶する特徴量記憶部82の内部には、第1の特徴量の種類ごとにデータが並べられている。図37の例では、入力ベクトルデータIの第1の特徴量は“2”と判断されたので、コードブック記憶部81と特徴量記憶部82の中から、第1の特徴量が“2”となる部分153のみが選択され、残りの部分は検索対象外となる。

## 【0271】

コードブック記憶部81および特徴量記憶部82の中で、以上のようにして第1の特徴量が“2”となる部分153に関してのみ、上記第12の実施形態と同様の手順でテンプレートベクトルデータの検索を行う。すなわち、入力ベクトルデータIの第2の特徴量を特徴量演算部85で演算し、その結果d4と、特徴量記憶部82内の特徴量d5とを用いて、特徴量差分演算部86ならびに演算省略判定部87により、コードブック記憶部81内の第1の特徴量が“2”となるテンプレートベクトルデータの中で、特にマンハッタン距離の演算が必要ないものを判断する。

## 【0272】

以上のように、第14の実施形態によれば、第12の実施形態で説明したように特徴量を単独で使う場合に比べて、より多くの演算を省略することができる。これは、第1の特徴量によってあらかじめ検索対象となるテンプレートベクトルデータを限定し、第2の特徴量で更に検索対象を絞り込むようにしているからである。これによって、特徴量を1つだけ使う場合に比べてより多くの演算の省略が可能となり、より高速に演算を行うことができる。

## 【0273】

また、テンプレートベクトルデータは、第1の特徴量だけでなく、第2の特徴量の大きさ順によっても並べることで2次元的な配列にしておくことにより、入力ベクトルデータIと第2の特徴量が似たテンプレートベクトルデータから検索を始めることができるので、入力ベクトルデータIと類似するテンプレートベクトルデータをより効率的に検索することができる。

なお、図37では、画像ブロックの大きさとして $4 \times 4$ 画素、つまりベクトルデータに直せば16次元のものを例に説明したが、この大きさは任意である。

#### 【0274】

##### (第15の実施形態)

図38は、第15の実施形態に係るデータ圧縮伸長システムの構成例を示す図である。本実施形態では、ベクトル量子化を利用したデータ圧縮伸長システムにおいて、データ圧縮・伸長時に使用するコードブックを保持しておくためのシステムを独立に用意しておくものである。

#### 【0275】

図38において、コードブック方式圧縮システム161は、外部より入力された、あるいは蓄積されているデータをコードブック方式により圧縮する装置である。コードブックサーバシステム162は、コードブック蓄積装置171、コードブック管理装置172、コードブック登録装置173、コードブック生成装置174を備え、圧縮・伸張システムからのリクエストに応じてコードブックを転送する機能を持つ。

#### 【0276】

コードブック方式圧縮システム161においてデータ圧縮の処理過程で使用するコードブックは、コードブックサーバ162よりネットワーク164を介して転送されたコードブック165を用いる。どのようなコードブックが必要であるかは、コードブックサーバ162に対するコードブック・リクエスト163をネットワーク164を介して送信する。

#### 【0277】

コードブックサーバ162は、コードブック・リクエスト163に対する返答として、以下のような動作を実行する。すなわち、コードブック管理装置172が、リクエストされたものに合ったコードブックのデータをコードブック蓄積装置171に蓄積されたコードブックの中から探し出し、それをネットワーク164を介してコードブック方式圧縮システム161に送信する。なお、コードブック蓄積装置171には、コードブック登録装置173によって事前に幾つかのコードブックが登録され、蓄積されている。

## 【0278】

上記コードブック蓄積装置171内にリクエストされたものに合致するコードブックがない場合には、最も似通ったコードブックからリクエストに合ったコードブックをコードブック生成装置174が生成する。このとき、コードブック蓄積装置171に蓄積されているコードブックを基に生成しても良いし、全く新たに生成しても良い。

## 【0279】

以上のようにして転送されたコードブック165を用いて、コードブック方式圧縮システム161で生成された圧縮データは、ネットワーク166を介してコードブック方式伸長システム167に送信される。

コードブック方式伸長システム167においてデータ伸長の処理過程で使用するコードブックは、コードブックサーバ162より転送されたコードブック170を用いる。どのようなコードブックが必要であるかは、コードブックサーバ162に対するコードブック・リクエスト168をネットワーク169を介して送信する。

## 【0280】

コードブック方式伸長システム167がリクエストする内容は、コードブック方式圧縮システム161で生成されたリクエスト内容を圧縮データと共に受け取り、これを使用しても良い。また、コードブック方式伸長システム167が独自に生成したリクエスト内容であっても良く、圧縮と伸長とで異なるコードブックを使用しても差し支えない。

## 【0281】

コードブックサーバ162は、コードブック・リクエスト168に対する返答として、以下のような動作を実行する。すなわち、コードブック管理装置172が、リクエストされたものに合ったコードブックのデータをコードブック蓄積装置171に蓄積されているコードブックの中から探し出し、それをネットワーク169を介してコードブック方式伸長システム167に送信する。

## 【0282】

リクエストされたものに合致するコードブックがない場合には、最も似通った

コードブックからリクエストに合ったコードブックをコードブック生成装置174が生成する。このとき、コードブック蓄積装置171に蓄積されているコードブックを基に生成しても良いし、全く新たに生成しても良い。

#### 【0283】

以上のように、本実施形態では、ベクトル量子化によるデータ圧縮・伸長時に使用するコードブックを保持しておくためのコードブックサーバ162を独立に用意しておく。さらに、このコードブックサーバ162と、データ圧縮システム161（データ送信側）と、データ伸長システム167（データ受信側）とを、それぞれ互いにアナログ電話回線、デジタル回線、インターネット専用線、通信回線等に代表されるネットワークを介して接続する。

#### 【0284】

そして、データ圧縮システム161、データ伸長システム167のそれぞれからコードブックの配信要求がコードブックサーバ162に与えられたときに、コードブックサーバ162がネットワークを介して瞬時にコードブックを送信するようとする。

#### 【0285】

これにより、各データ圧縮・伸長システムがコードブックを保持する必要がなく、かつ、コードブックが更新された場合にはコードブックサーバ162のデータを更新するだけで良いので、非常に簡単なデータ転送システムを実現することができる。また、常に最新のコードブックをコードブックサーバ162から受け取ることができ、データの質を良好に保つことができる。

#### 【0286】

##### （第16の実施形態）

第16の実施形態では、各フレーム毎にフレーム内でのベクトル量子化（空間方向ベクトル量子化）を行った後で、それにより得られたコード番号を並べ替えてフレーム間でのベクトル量子化（時間軸方向ベクトル量子化）を行う上述の実施形態を応用した例について、以下に述べる。

#### 【0287】

これまでに述べた第2～第7の実施形態では、それぞれのフレームから空間方

向ベクトル量子化後のコード番号を1つずつ抜き出してきて、それらをフレーム画像順に並べ替えることによって新たなベクトル（時間軸ベクトル）を生成していた。これに対して、第16の実施形態では、1枚のフレームから抜き出してくるコード番号は1つではなく、各フレームからそれぞれ複数のコード番号を抜き出してきて新たな時間軸ベクトルを生成するものである。

#### 【0288】

図39は、本実施形態に係る空間方向コード番号の並べ替え方を説明するためのデータフロー図である。図39に示すように、まず原画像のそれぞれのフレームに対して、 $4 \times 4$ 画素単位のマクロブロック毎に空間方向ベクトル量子化(SDVQ)の処理を行い、各マクロブロックを空間方向用コードベクトルに対応するコード番号で置き換える。図39の例では、あるフレームの右隅のマクロブロックがコード番号“30”に置き換えられ、その周りの3つのマクロブロックがコード番号“15”、“10”、“20”に置き換えられていることが示されている。

#### 【0289】

このような空間方向ベクトル量子化の処理が各フレーム（図39の例では4つのフレーム）について行われると、空間方向用コードブックのコード番号で表現されたそれぞれのフレーム画像中から、同じアドレスが示すブロックのコード番号を $2 \times 2$ のブロック単位で4個ずつ取り出してきて、それらをフレーム画像順に並べ替える。これにより、時間軸方向に分散して存在する同アドレス4ブロック分の各コード番号を1つの時間軸ベクトルのデータ列、つまり時間軸方向ベクトル量子化(TDVQ)用の1つのマクロブロックとしてまとめて表現する。

#### 【0290】

図40は、本実施形態による空間方向コード番号の再配列の仕方を上述した第2の実施形態に応用した場合の圧縮時のデータの流れを示すデータフロー図であり、また図41および図42は、その場合のデータ圧縮伸長システムの構成例を示すブロック図である。なお、図40に関して、圧縮対象とするフレーム数が4フレームであること、およびコード番号の再配列の仕方が異なること以外は図11に示した第2の実施形態と同じなので、ここでは説明を省略する。

## 【0291】

図41において、本実施形態のデータ圧縮システムでは、図9に示した第2の実施形態によるデータ圧縮システムに対して、空間方向コード再配列方法指定部181が更に追加されている。また、図42において、本実施形態のデータ伸長システムでは、図10に示した第2の実施形態によるデータ伸長システムに対して、空間方向コード再生方法指定部182が更に追加されている。

## 【0292】

空間方向コード再配列方法指定部181は、空間方向コード番号を再配列して新たな時間軸ベクトルを生成する際に、1枚のフレームから抜き出してくるコード番号を1個とするか、それとも各フレームからそれぞれ4個のコード番号を抜き出してくるかを指定するものである。言い換えれば、ベクトル量子化の一連の処理を図11の流れに従って行うか、それとも図40の流れに従って行うかを指定するものである。

## 【0293】

また、空間方向コード再生方法指定部182は、これとは逆に、時間軸コード番号に対応するコードベクトルを時間軸コードブックから取り出す際に、1つの時間軸コード番号に対して1個のコードベクトルを取り出すか、それとも4個のコードベクトルを取り出すかを指定するものである。この指定は、空間方向コード再配列方法指定部181でどちらの再配列方法が指定されたかに応じて、対応する再生方法が指定される。

## 【0294】

図43および図44は、本実施形態による空間方向コード番号の再配列の仕方を上述した第3の実施形態および第4の実施形態に応用した場合の圧縮時のデータの流れを示すデータフロー図である。これらの図43および図44に関して、圧縮対象とするフレーム数が5フレームであること、およびコード番号の再配列の仕方が異なること以外は、図12や図13に示した第3の実施形態および第4の実施形態と同じなので、ここでは説明を省略する。

## 【0295】

また、図45は、本実施形態による空間方向コード番号の再配列の仕方を上述

した第7の実施形態に応用した場合のデータ圧縮システムの構成例を示すブロック図である。図45において、本実施形態のデータ圧縮システムでは、図15に示した第7の実施形態によるデータ圧縮システムに対して、空間方向コード再配列方法指定部181および空間方向DC成分再配列方法指定部183が更に追加されている。

#### 【0296】

上記空間方向コード再配列方法指定部181は、上記図41に示したものと同じものである。また、空間方向DC成分再配列方法指定部183は、各ブロックの最小輝度値のみで表現された各フレームの画像に対して、同じアドレスが示すブロックのデータ値をフレーム画像順に再配列して新たな時間軸ベクトルを生成する際に、1枚のフレームから抜き出してくるDC成分を1個とするか、それとも各フレームからそれぞれ4個のDC成分を抜き出してくるかを指定するものである。

#### 【0297】

以上のように、第16の実施形態では、空間方向コード番号やDC成分の再配列の仕方を指定した方法で行うように構成し、各フレームからそれぞれ複数のコード番号やDC成分を抜き出してきて新たな時間軸ベクトルを生成できるようにすることにより、ある程度空間方向にまとまったブロックを単位として時間軸ベクトル量子化を行うことができるので、高い圧縮率を維持しつつ再生画像の品質を更に向上させることができる。

#### 【0298】

なお、以上に示した例では、4フレームのそれぞれから空間方向コード番号を $2 \times 2$ のブロック単位で4個ずつ取り出して時間軸ベクトルの1つのマクロブロックとしたが、4フレームのそれぞれから $3 \times 3$ のブロック単位で9個ずつ取り出すようにしても良い。また、この他にも、空間方向コード番号のマクロブロックの取り方や圧縮対象のフレーム数を任意に設定することが可能である。

#### 【0299】

図46に、様々に設定した空間方向コード番号の再配列の方法と、そのときの圧縮率との関係を示す。これから分かるように、図46に示される表の下の方の

設定ほど、高い圧縮率を達成できている。

### 【0300】

#### (その他の実施形態)

上記様々な実施形態に示した各機能ブロックおよび処理手順は、ハードウェアにより構成しても良いし、CPUあるいはMPU、ROMおよびRAM等からなるマイクロコンピュータシステムによって構成し、その動作をROMやRAMに格納された作業プログラムに従って実現するようにしても良い。また、上記各機能ブロックの機能を実現するように当該機能を実現するためのソフトウェアのプログラムをRAMに供給し、そのプログラムに従って上記各機能ブロックを動作させることによって実施したものも、本発明の範疇に含まれる。

### 【0301】

この場合、上記ソフトウェアのプログラム自体が上述した各実施形態の機能を実現することになり、そのプログラム自体、およびそのプログラムをコンピュータに供給するための手段、例えばかかるプログラムを格納した記録媒体は本発明を構成する。かかるプログラムを記憶する記録媒体としては、上記ROMやRAMの他に、例えばフロッピーディスク、ハードディスク、光ディスク、光磁気ディスク、CD-ROM、CD-I、CD-R、CD-RW、DVD、zip、磁気テープ、あるいは不揮発性のメモリカード等を用いることができる。

### 【0302】

また、コンピュータが供給されたプログラムを実行することにより、上述の実施形態の機能が実現されるだけでなく、そのプログラムがコンピュータにおいて稼働しているOS（オペレーティングシステム）あるいは他のアプリケーションソフト等の共同して上述の実施形態の機能が実現される場合にもかかるプログラムは本発明の実施形態に含まれることは言うまでもない。

### 【0303】

さらに、供給されたプログラムがコンピュータの機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに格納された後、そのプログラムの指示に基づいてその機能拡張ボードや機能拡張ユニットに備わるCPU等が実際の処理の一部または全部を行い、その処理によって上述した実施形態の機能

が実現される場合にも本発明に含まれることは言うまでもない。

#### 【0304】

##### 【発明の効果】

以上詳しく説明したように、第1の本発明によれば、ベクトル量子化を行う際に使用するベクトルを構成するブロックを、データ位置を1次元的に配列したラインブロックとすることとしたので、例えば圧縮対象として画像データを用いた場合、ラインブロック内のデータ配列を再生装置のスキャン方向と合致させることができ、従来のマクロブロックの場合に必要だった伸長・再生時のアドレス変換操作を回避することができる。これにより、再生画像の表示に要する時間を大幅に短縮することができる。

#### 【0305】

また、第2、第3の発明によれば、圧縮時にベクトル量子化を行う際に使用するベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして圧縮対象より抽出するか、もしくは、伸長時にコードブックの中から探し出したコードベクトルを構成するブロックを、少なくとも垂直方向に近接するブロック間では空間的位置を水平方向にずらして割り当てるようにしたので、例えば圧縮対象として画像データを用いた場合、再生画像上において各ブロック境界の連續性を断つてブロック境界（量子化誤差）を目立ちにくくすることができる。これにより、圧縮率を保ったまま質の高い再生画像を得ることができる。

#### 【0306】

また、第4～第6の発明によれば、時間軸に沿って変化する圧縮対象をベクトル量子化する際に、同一時間内の1つのデータ列に対して空間方向に対するベクトル量子化を行うだけでなく、時間軸に沿った複数のデータ列間で時間軸方向に対するベクトル量子化も行うようにしたので、例えば、動画像の各フレーム内でデータ圧縮を行うだけでなく、各フレーム間に対してもデータ圧縮を行うことができる。これにより、再生画像の品質を維持したままより高い圧縮率で動画を圧縮することができる。

#### 【0307】

また、第7の発明によれば、圧縮時において時間軸方向に対するベクトル量子化を行うために複数の新たなベクトルを生成する際に、上記新たなベクトル内の要素を少なくとも近接するベクトル間では時間軸方向にずらして配列するようするか、もしくは、伸長時に上記圧縮時とは逆の処理を行う際に複数の新たなベクトルを生成する場合に、上記新たなベクトル内の少なくとも近接する要素を時間軸方向にずらして配列するようにしたので、例えば圧縮対象として動画を用いた場合、伸長時に各時間毎の画像を再生していく毎に目立っていた量子化誤差を目立たなくすることができ、動画の高い圧縮率を保ったまま品質の良い再生画像を得ることができる。

#### 【0308】

また、第8の発明によれば、圧縮対象から少なくとも1つ以上の特徴量を抽出して特徴量データと残りの基本パターンデータとに分離し、分離した特徴量データと基本パターンデータとに対してそれぞれ独立にベクトル量子化を行うようにしたので、様々な特徴が絡み合った圧縮対象そのものに対するコードブックに比べて、それぞれのデータのベクトル量子化で使用するコードブックを単純化することができ、必要なパターン数も減らすことができる。これにより、ベクトル量子化により類似度の大きいパターンをコードブック中から探し出す際に、より適切なパターンに絞り込みやすくなり、高精度のベクトル量子化を行うことができる。したがって、例えば圧縮対象として画像データを用いた場合、高い圧縮率を保ったままより高品位な再生画像を得ることができる。

#### 【0309】

また、第9の発明によれば、時間軸に沿って変化する圧縮対象の各時刻におけるデータに対してそれぞれベクトル量子化を実行し、これにより各時刻毎に得られるコード列について、時間軸方向の相関が小さい部分のコードのみを出力するようにしたので、例えば圧縮対象として動画を用いた場合、フレーム間でデータ値が大きく変化したところのデータだけを転送し、変化が少ないとところのデータは転送しないようにすることができる。これにより、個々のフレーム内で空間方向に対してベクトル量子化をした結果に加えて、フレーム間に対してもデータを圧縮することができ、高い圧縮率で動画を圧縮・転送することができる。

## 【0310】

また、第10の発明によれば、カラー画像をベクトル量子化する場合に、色信号用のコードブックよりも輝度信号用のコードブックの方により多くのコードベクトルを割り当てるようにしたので、画質にとってより重要度の大きい輝度信号に関してより類似度の高いコードベクトルをベクトル量子化によって得ることができる。したがって、全体としてのコード長を変えないようにすることで、圧縮率を維持したまま再生画像の画質を向上させることができる。

## 【0311】

また、第11の発明によれば、コードブックの作成する際に、初期のコードブックとして、とり得る値の最小値から最大値までデータ値が連続的に変化するパターンのコードブックを用い、1回の処理で更新する範囲を仮想的な2次元平面上で1次元的に適用し、かつ更新回数の増加と共に範囲を減少させ、更新係数の初期値を0.3～1の範囲内の何れかの値とし、かつ更新回数の増加と共に更新係数の値を減少させるようにしたので、コードブックの最適化をより効率的に行なうことができるとともに、高精度なコードブックを得ることができる。これにより、生成されたコードブックを例えば画像のデータ圧縮伸長システムに用いることで、再生画像の画質を向上させることができる。

## 【0312】

また、第12の発明によれば、コードブックを作成する際に、複数のサンプルデータに対して独立にコードブックを最適化し、得られた複数のコードブックを合成して新たなコードブックを作成するようにしたので、様々な画像に対して耐性を有するコードブックを作成することができる。これにより、コードブックを記憶するための装置のサイズを大きくすることなく、汎用的なコードブックを得ることができる。

## 【0313】

また、第13の発明によれば、圧縮対象より抽出されるベクトルの特徴量と、コードブック内の各コードベクトルの特徴量とを比較し、その比較結果に基づいて、各コードベクトルのそれぞれについて上記圧縮対象ベクトルとの類似度を求める演算を省略するかどうかを決定するようにしたので、類似度の演算に比べて

演算量の少ない特微量の演算を行うことによって適宜類似度の演算を省略することができ、圧縮対象ベクトルに類似したコードベクトルを検索する操作を行う際に、その検索に必要な演算量を少なくすることができ、ベクトル量子化の演算を全体として高速に行うことができる。

#### 【0314】

また、第14の発明によれば、ベクトル量子化により圧縮伸長を行う際に必ず必要となるコードブックを圧縮・伸長システムとは別に用意するようにしたので、各圧縮・伸長システムはコードブックを保持する必要がなく、実際のデータのみを転送するだけで良くなり、より容量の少ない媒体を使ってデータ転送を行うことができる。これにより、非常に簡単なデータ転送システムを実現することができる。

#### 【図面の簡単な説明】

##### 【図1】

第1の実施形態によるデータ圧縮装置の構成を示すブロック図である。

##### 【図2】

第1の実施形態によるデータ伸長装置の構成を示すブロック図である。

##### 【図3】

第1の実施形態によるデータ圧縮装置の動作を示すフローチャートである。

##### 【図4】

第1の実施形態によるデータ伸長装置の動作を示すフローチャートである。

##### 【図5】

ラインブロックをベクトルデータとして用いた第1の実施形態による特徴を説明するための図である。

##### 【図6】

ラインブロックを貼り込む位置をずらした第1の実施形態による特徴を説明するための図である。

##### 【図7】

640×480画素の1枚の画像を圧縮・伸長処理するために要する時間を示す図である。

【図8】

ベクトル量子化システム全体のデータの流れを説明するための図である。

【図9】

第2～第4の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図10】

第2～第4の実施形態によるデータ伸長装置の構成を示すブロック図である。

【図11】

第2の実施形態による圧縮時のデータの流れを示すデータフロー図である。

【図12】

第3の実施形態による圧縮時のデータの流れを示すデータフロー図である。

【図13】

第4の実施形態による圧縮時のデータの流れを示すデータフロー図である。

【図14】

第5の実施形態を説明するためのブロック図である。

【図15】

第7の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図16】

第7の実施形態による圧縮時のデータの流れを示すデータフロー図である。

【図17】

第8の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図18】

第8の実施形態によるデータ圧縮装置をソフトウェアにより実現する場合の構成を示すブロック図である。

【図19】

第8の実施形態によるデータ圧縮装置の動作を示すフローチャートである。

【図20】

第9の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図21】

第10の実施形態によるコードブック作成方法を説明するための図である。

【図22】

第10の実施形態を示す図であり、初期のコードブックを様々に与えた場合のP S N R特性の違いを説明するための図である。

【図23】

第10の実施形態を示す図であり、書き換えパターンを様々に変えた場合のP S N R特性の違いを説明するための図である。

【図24】

第10の実施形態を示す図であり、利得の与えた方を様々に変えた場合のP S N R特性の違いを説明するための図である。

【図25】

第10の実施形態を示す図であり、利得の初期値を様々に与えた場合のP S N R特性の違いを説明するための図である。

【図26】

第11の実施形態について説明するための図である。

【図27】

第11の実施形態について説明するためのP S N R特性図である。

【図28】

第12の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図29】

第12の実施形態によるデータ圧縮装置の動作を示すフローチャートである。

【図30】

第13の実施形態について説明するための図である。

【図31】

第13の実施形態について説明するための特性図である。

【図32】

第14の実施形態について説明するための図である。

【図33】

第14の実施形態について説明するための図である。

【図34】

第14の実施形態について説明するための図である。

【図35】

第14の実施形態について説明するための図である。

【図36】

第14の実施形態について説明するための図である。

【図37】

第14の実施形態によるデータ圧縮装置の構成を示すブロック図である。

【図38】

第15の実施形態によるデータ圧縮伸長システムの構成を示すブロック図である。

【図39】

第16の実施形態による空間方向コード番号の並べ替え方を説明するためのデータフロー図である。

【図40】

第16の実施形態による空間方向コード番号の再配列の仕方を第2の実施形態に応用した場合の圧縮時のデータの流れを示すデータフロー図である。

【図41】

第16の実施形態によるデータ圧縮装置の構成例を示すブロック図である。

【図42】

第16の実施形態によるデータ伸長装置の構成例を示すブロック図である。

【図43】

第16の実施形態による空間方向コード番号の再配列の仕方を第3の実施形態に応用した場合の圧縮時のデータの流れを示すデータフロー図である。

【図44】

第16の実施形態による空間方向コード番号の再配列の仕方を第4の実施形態に応用した場合の圧縮時のデータの流れを示すデータフロー図である。

【図45】

第16の実施形態による空間方向コード番号の再配列の仕方を第7の実施形態に応用した場合のデータ圧縮装置の構成例を示すブロック図である。

## 【図46】

空間方向コード番号の再配列の方法と圧縮率との関係を示す図である。

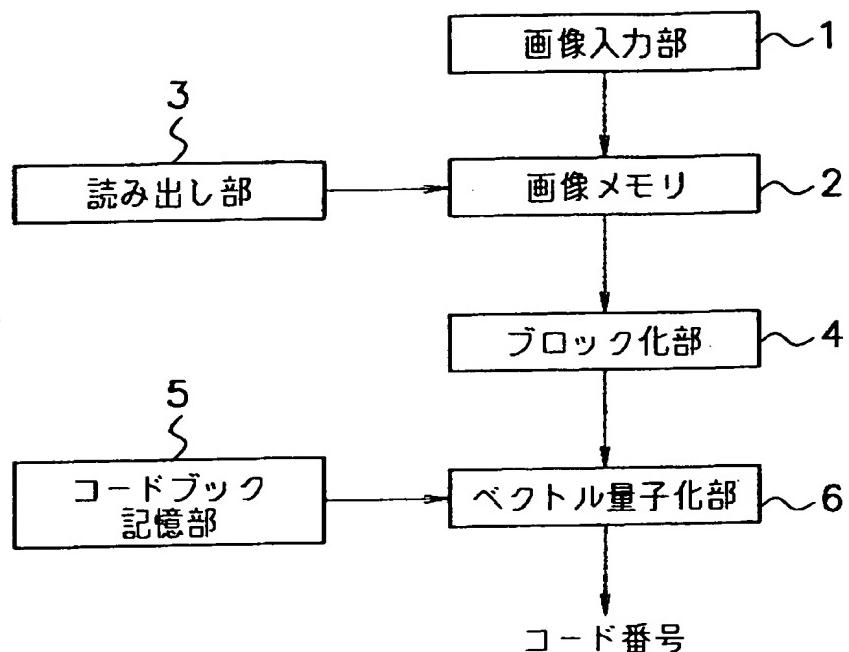
## 【符号の説明】

- 1 画像入力部
- 2 画像メモリ
- 3 読み出し部
- 4 ブロック化部
  - 1 1 再生画像生成部
  - 1 2 ブロックシフト部
- 3 5 空間方向用コードブック記憶部
- 3 6 相関度演算部
- 3 7 演算モード指定部
- 3 8 コード決定部
- 3 9 空間方向コード再配列演算部
- 4 0 時間軸用コードブック記憶部
- 4 4 空間方向コード再生部
- 4 5 時間軸用コードブック記憶部
- 4 8 再生画像生成部
- 4 8 a 時間軸シフト部
- 4 9 空間方向用コードブック記憶部
- 5 1 DC成分検出・除去部
- 5 2 演算モード指定部
- 5 3 空間方向用コードブック記憶部
- 5 4 空間方向DC成分再配列演算部
- 5 5 時間軸用コードブック記憶部
- 5 6 時間軸DC成分用コードブック記憶部
- 6 4 コード列記憶部
- 6 5 コード列比較部
- 6 6 出力部

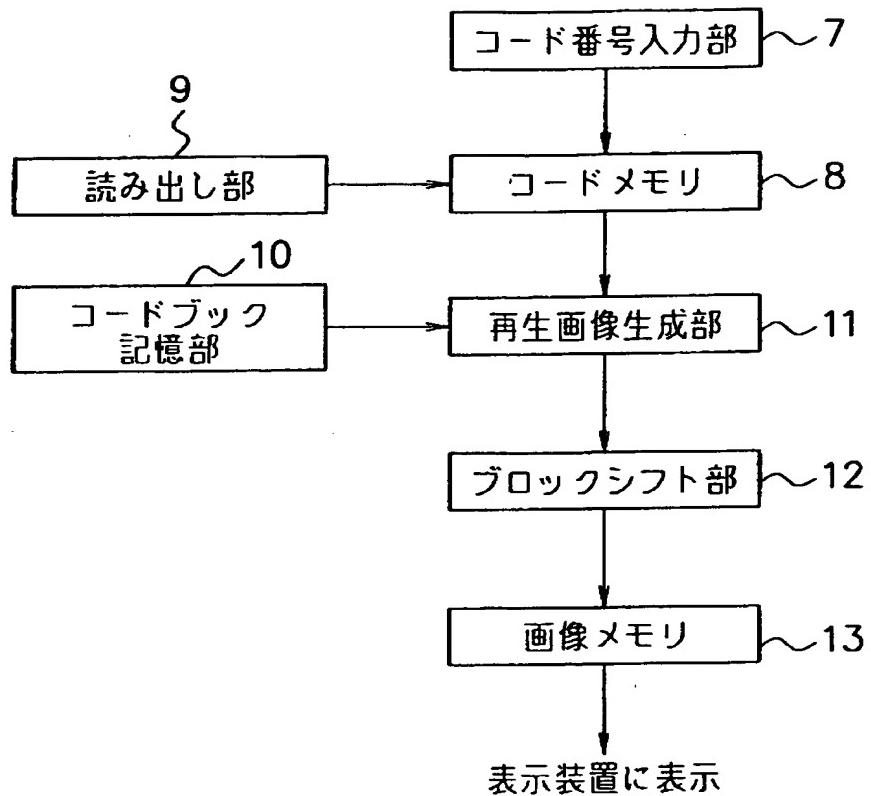
- 7 4 Y, U, V信号分離部
- 7 5 ブロック化部
- 7 6 空間方向用コードブック記憶部
- 7 7 相関度演算部
- 7 8 コード決定部
- 8 1 コードブック記憶部
- 8 2 特微量記憶部
- 8 3 マンハッタン距離演算部
- 8 4 最小マンハッタン距離記憶部
- 8 5 特微量演算部
- 8 6 特微量差分演算部
- 8 7 演算省略判定部
- 8 8 アドレスカウンタ
- 8 9 最小距離アドレス記憶部
- 1 5 1 変化モード判定部
- 1 5 2 特徴テンプレート記憶部
- 1 6 2 コードブックサーバシステム
- 1 7 1 コードブック蓄積装置
- 1 7 2 コードブック管理装置
- 1 7 4 コードブック生成装置
- 1 8 1 空間方向コード再配列方法指定部
- 1 8 2 空間方向コード再生方法指定部
- 1 8 3 空間方向DC成分再配列方法指定部

【書類名】 図面

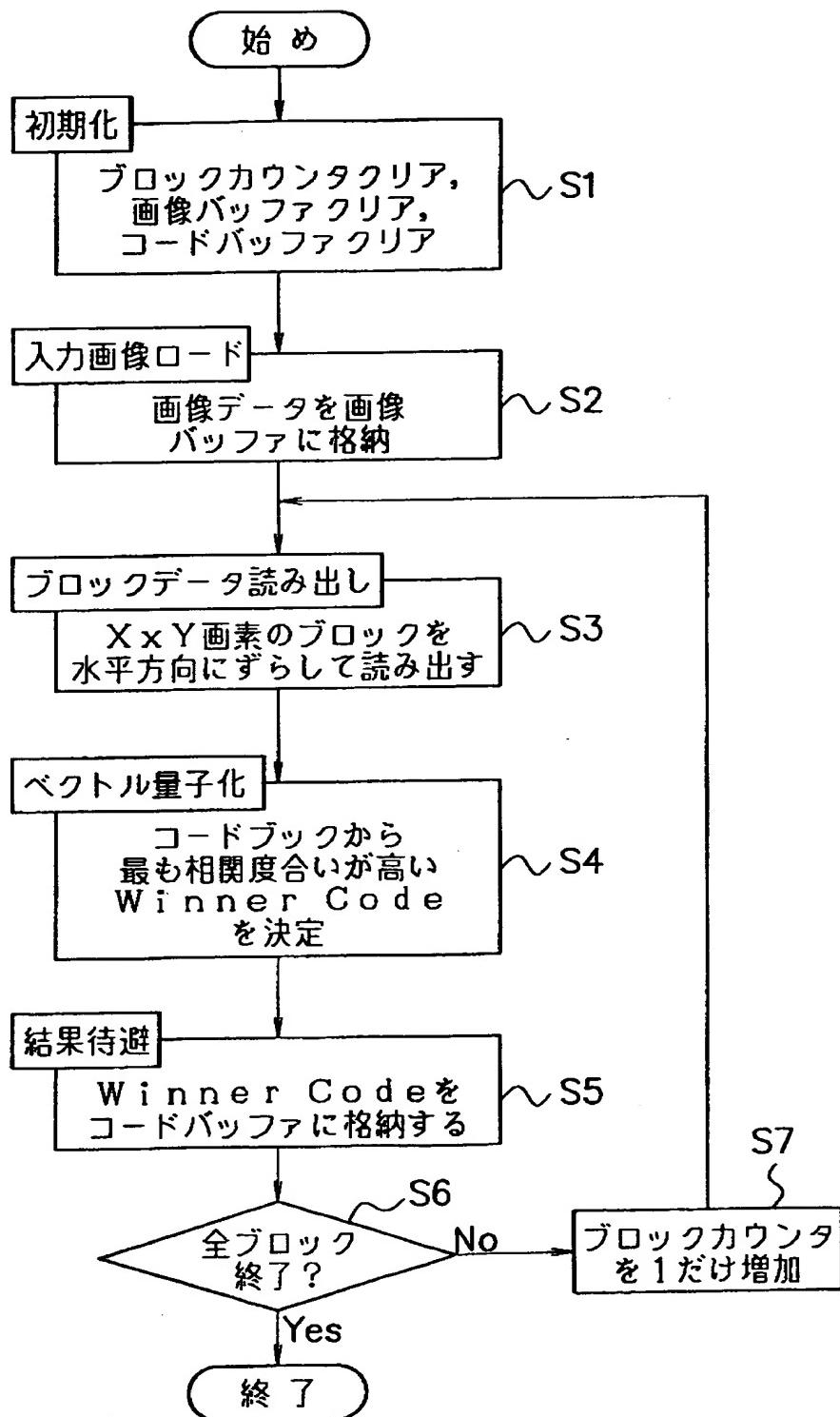
【図1】



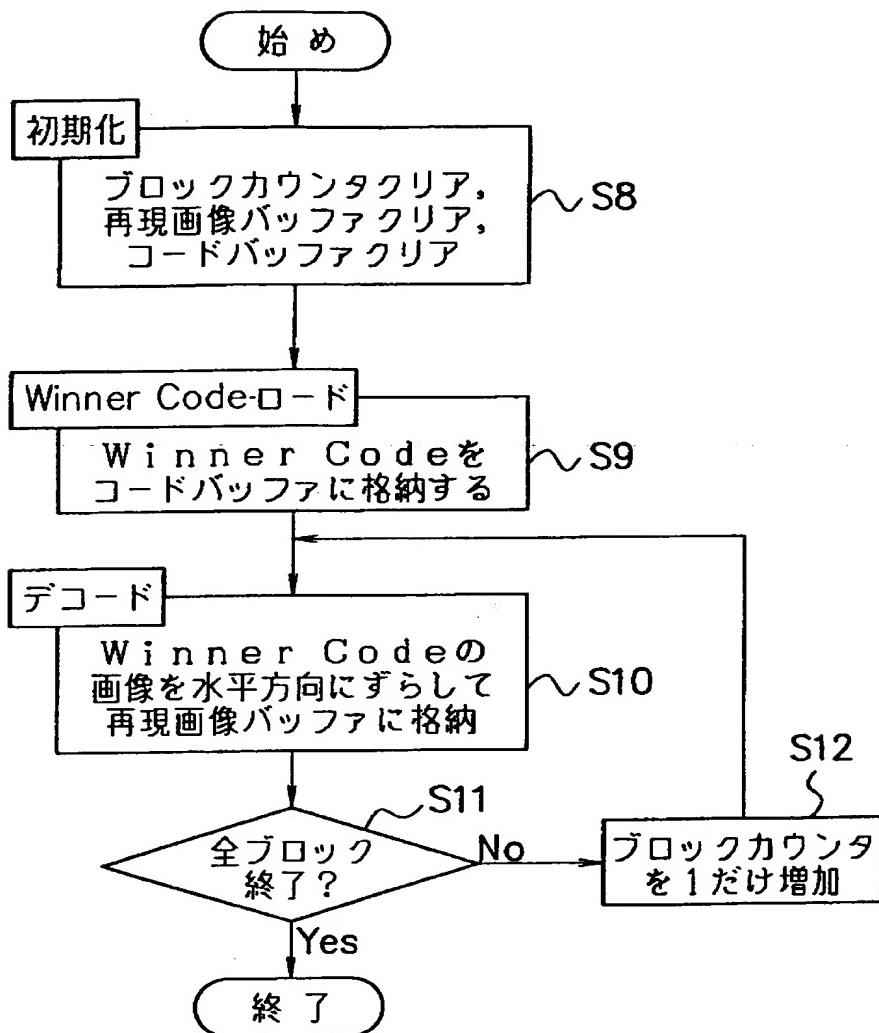
【図2】



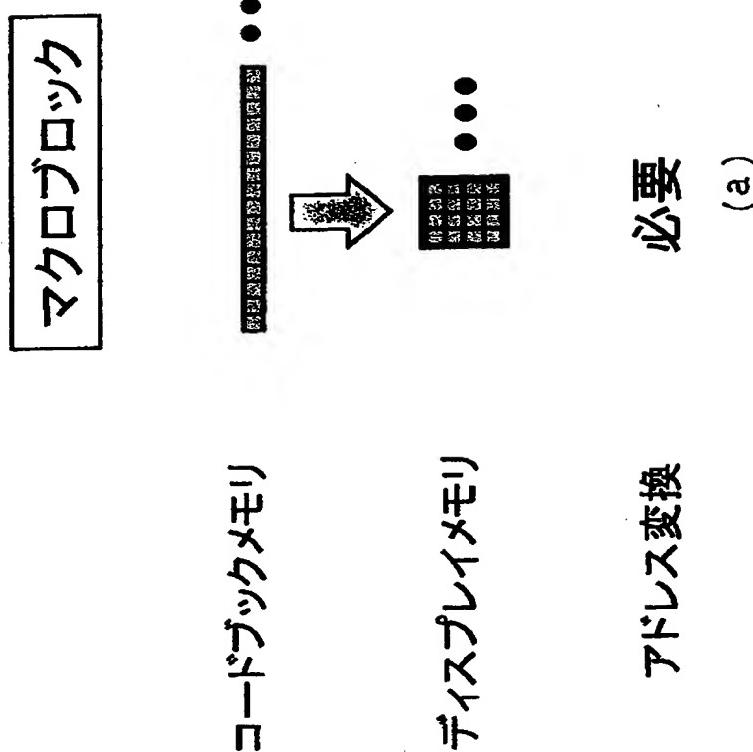
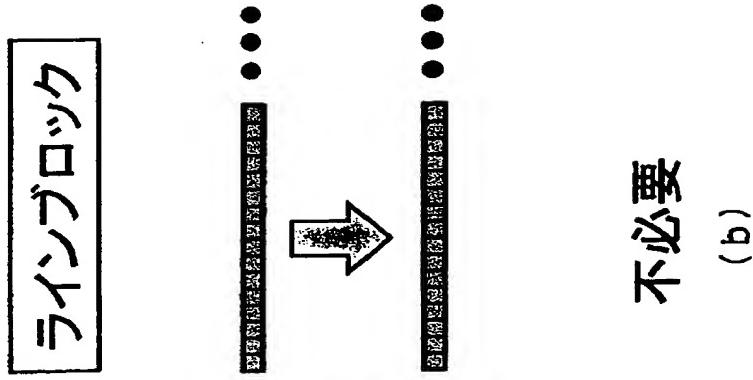
【図3】



【図4】

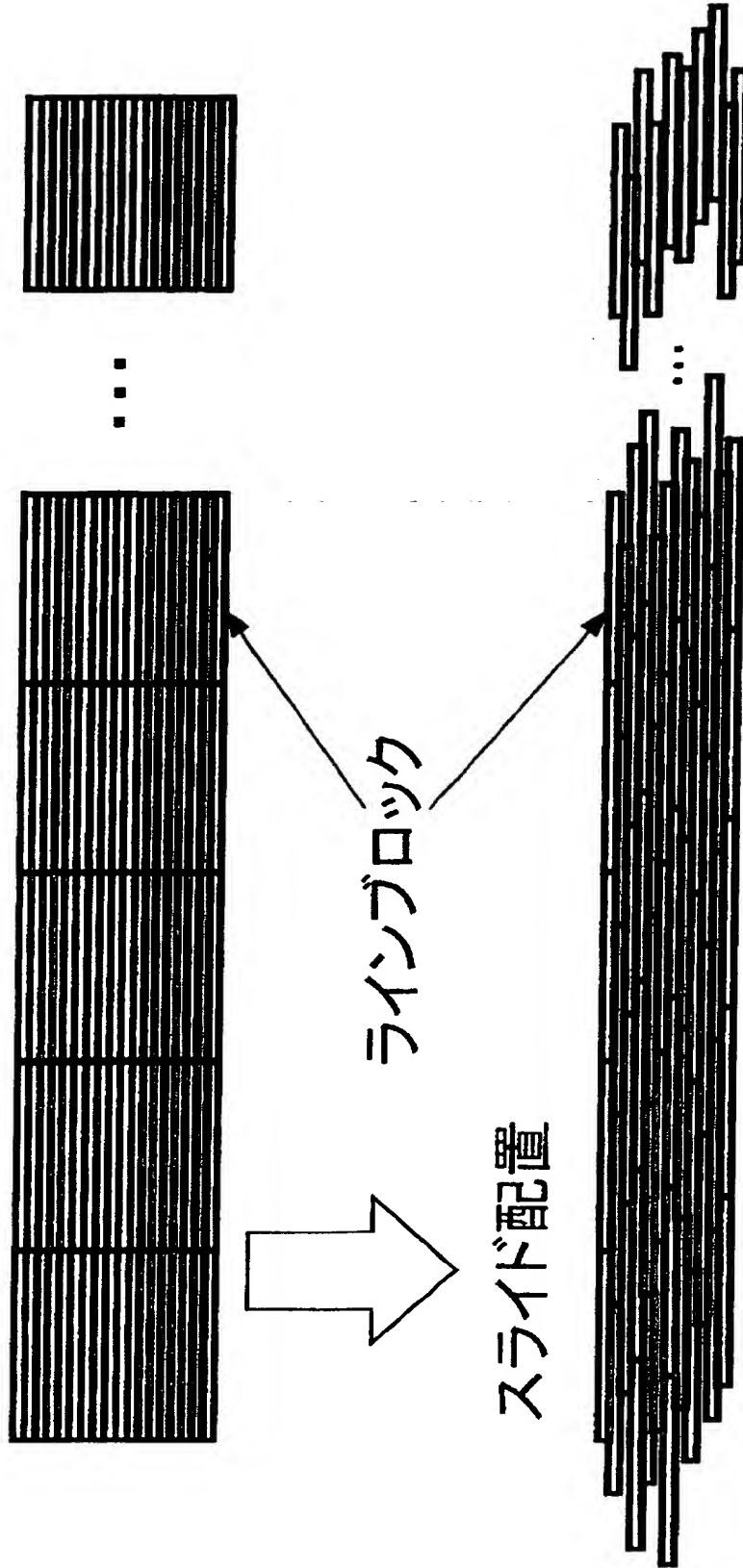


【図5】

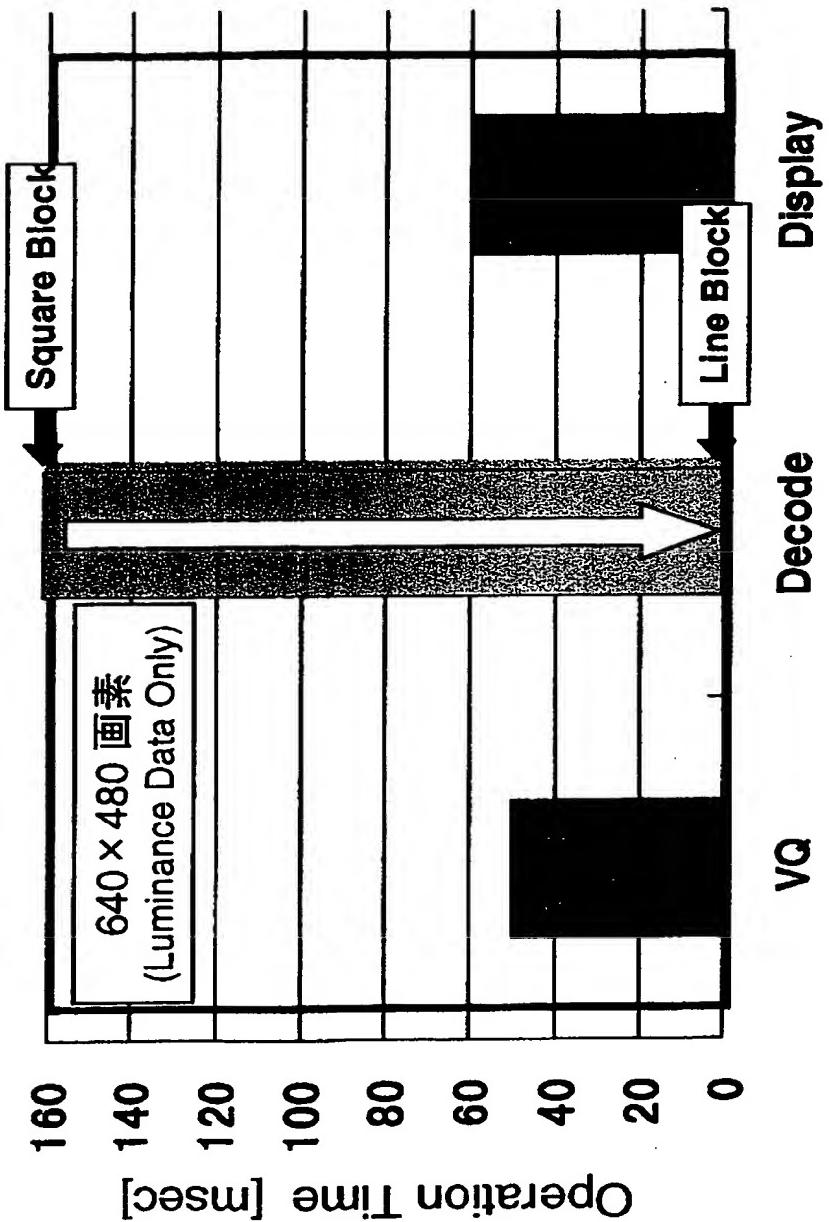


【図6】

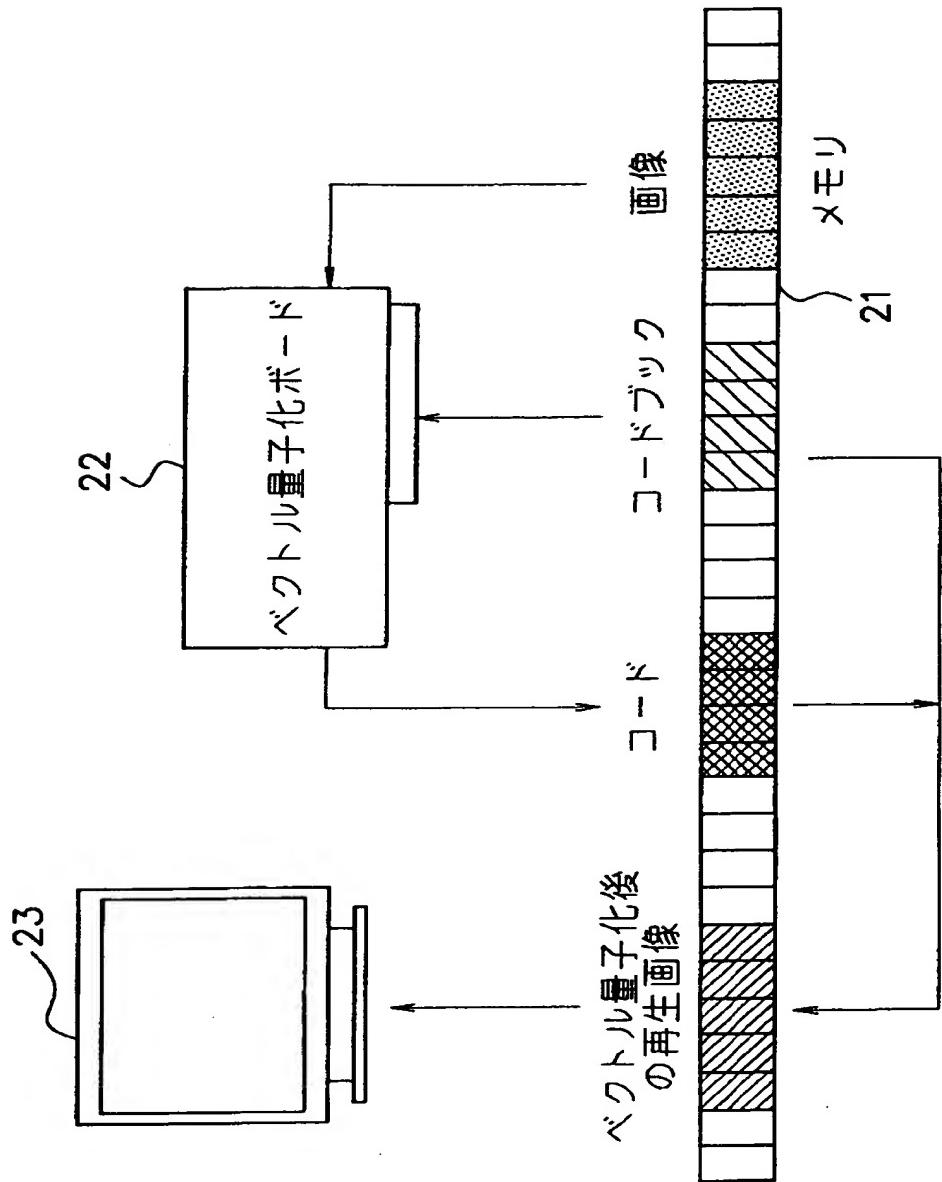
通常配置



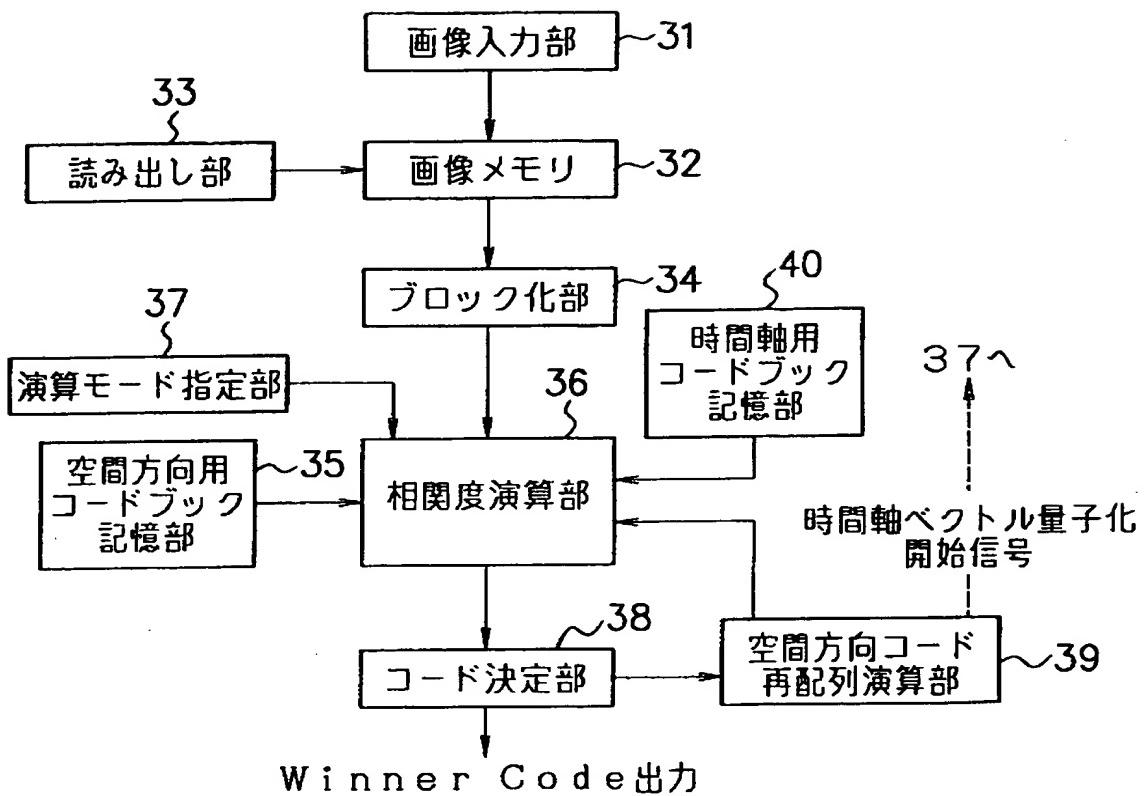
【図 7】



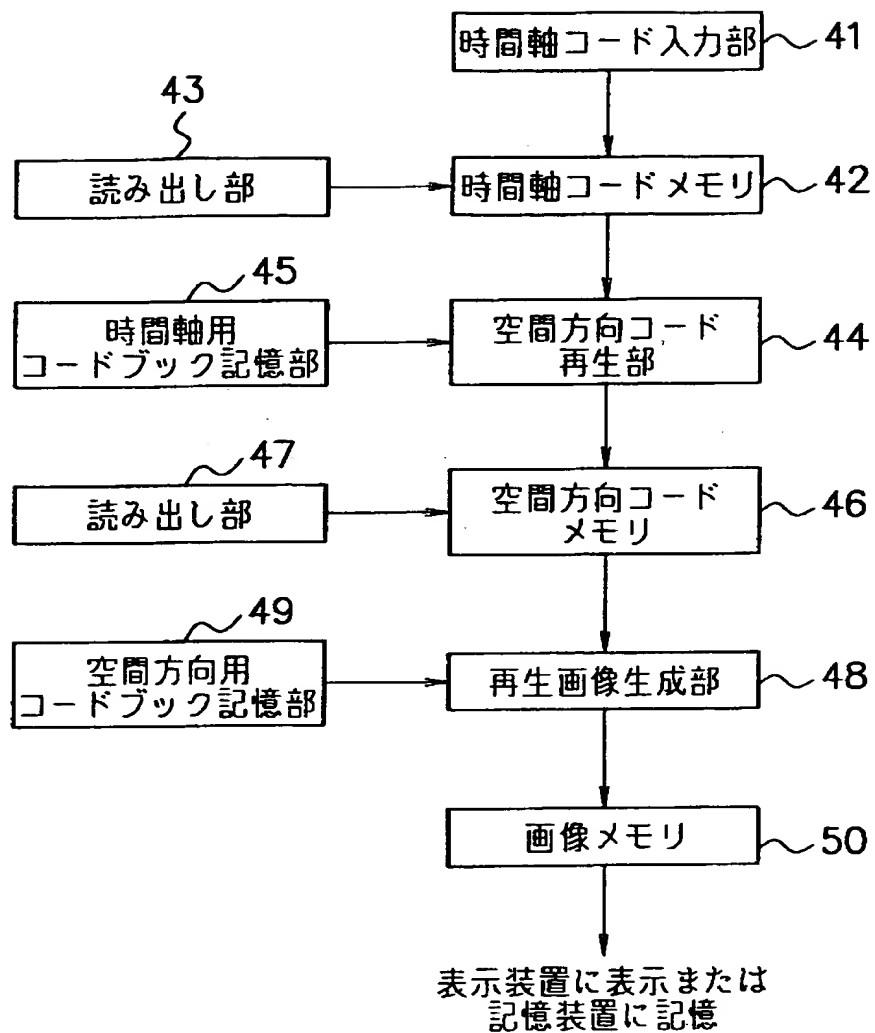
【図8】



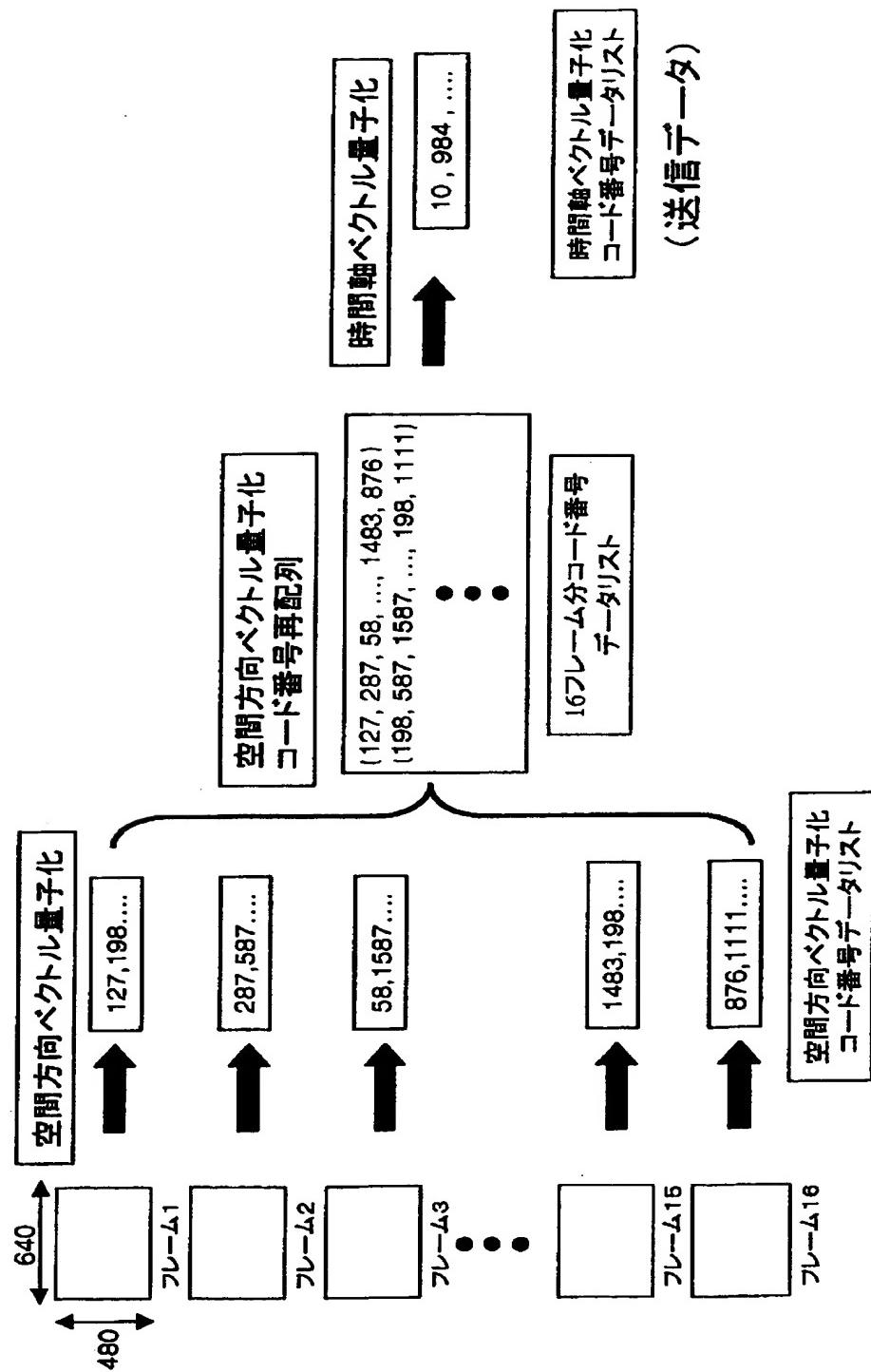
【図9】



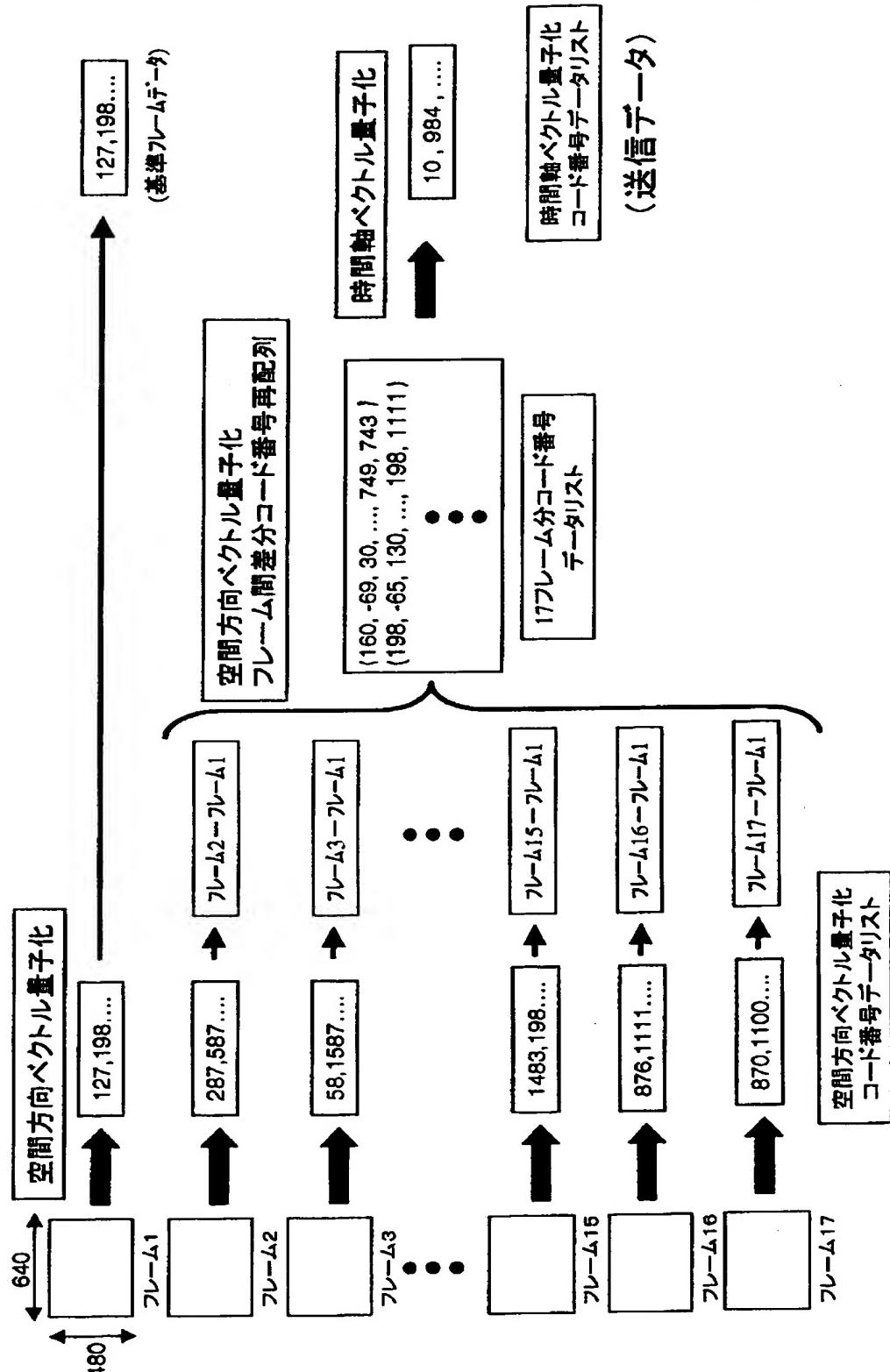
【図10】



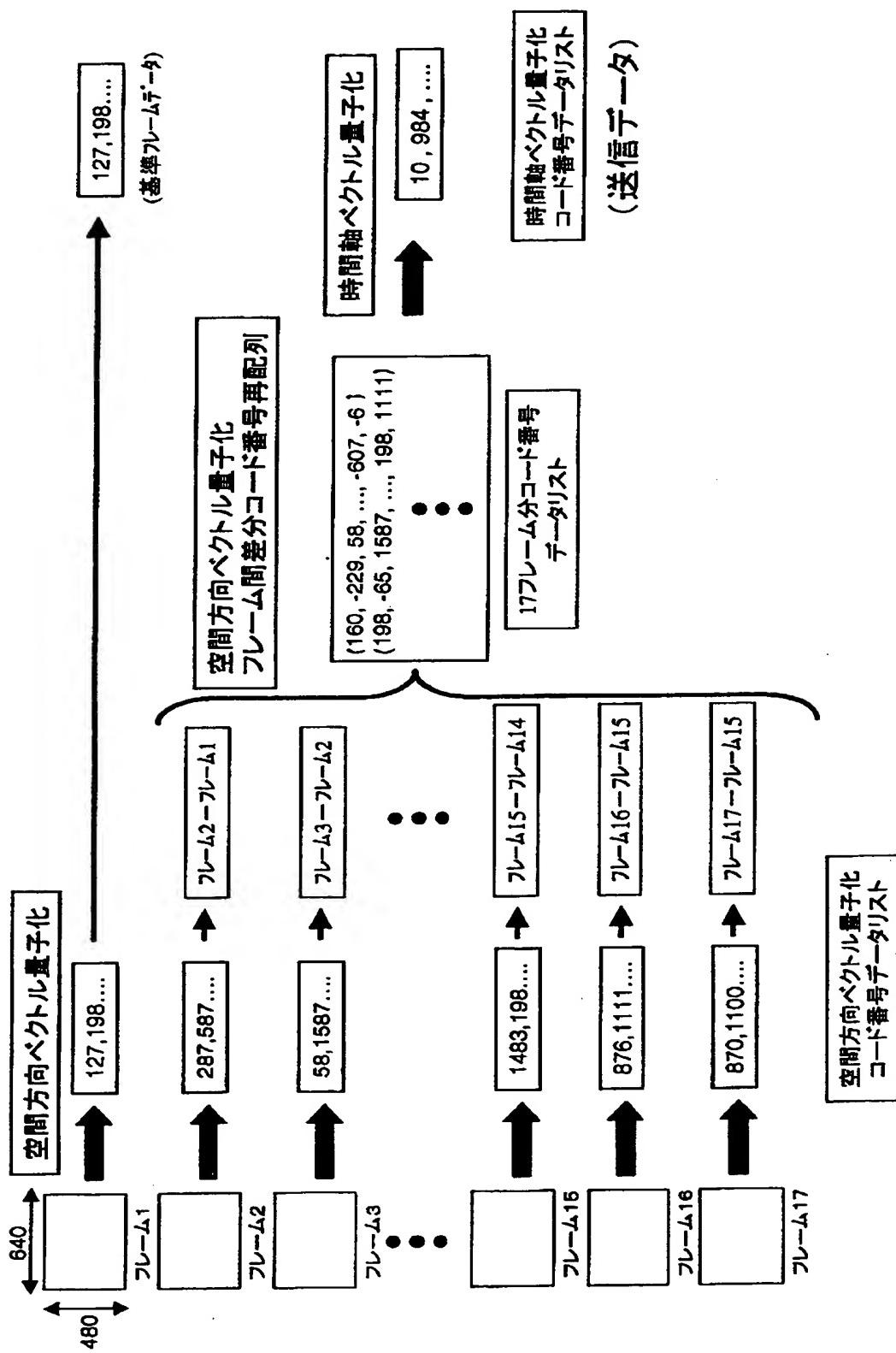
【図11】



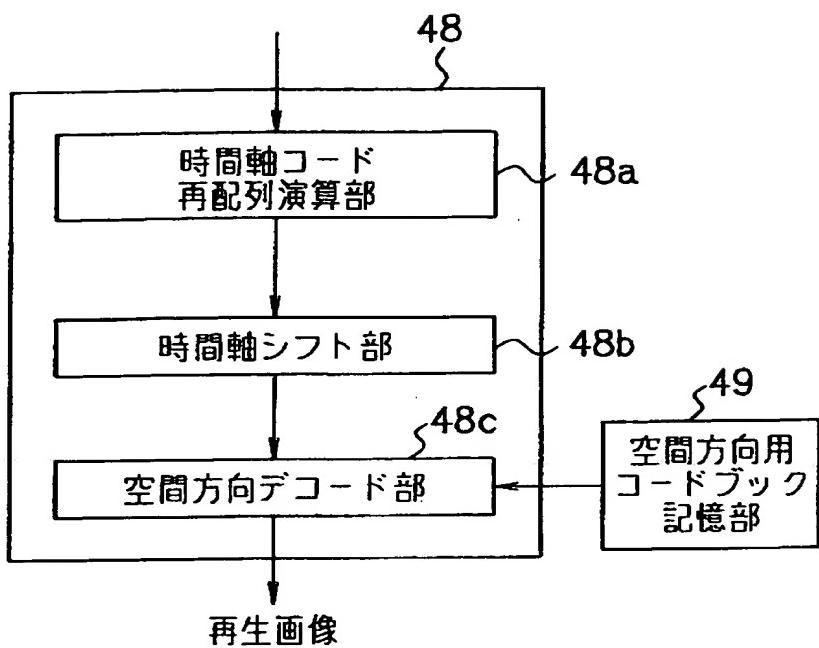
【図12】



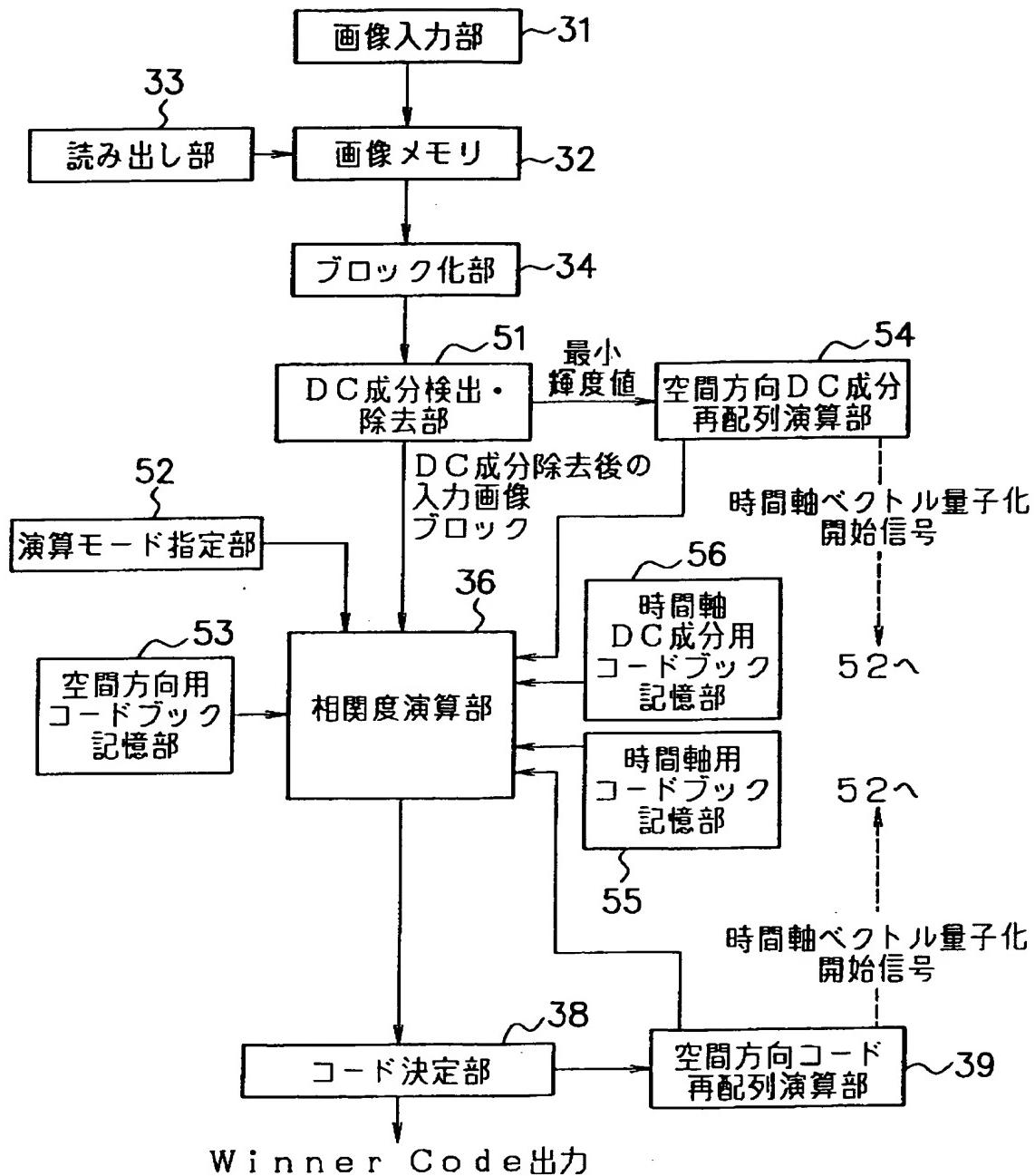
【図13】



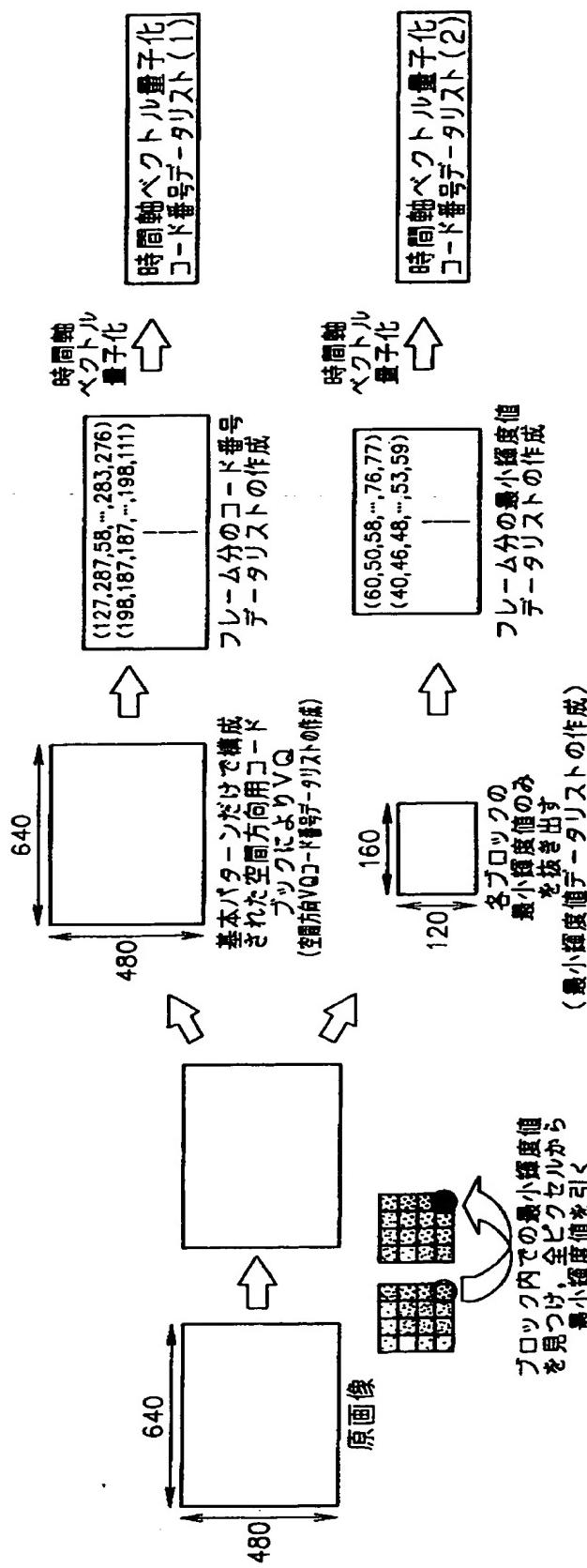
【図14】



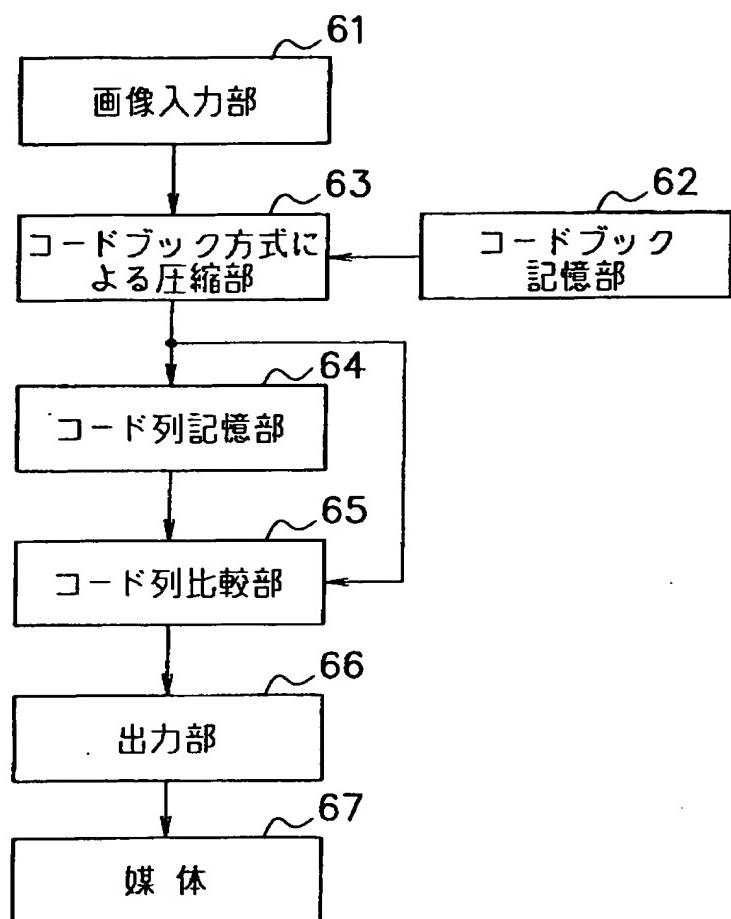
【図15】



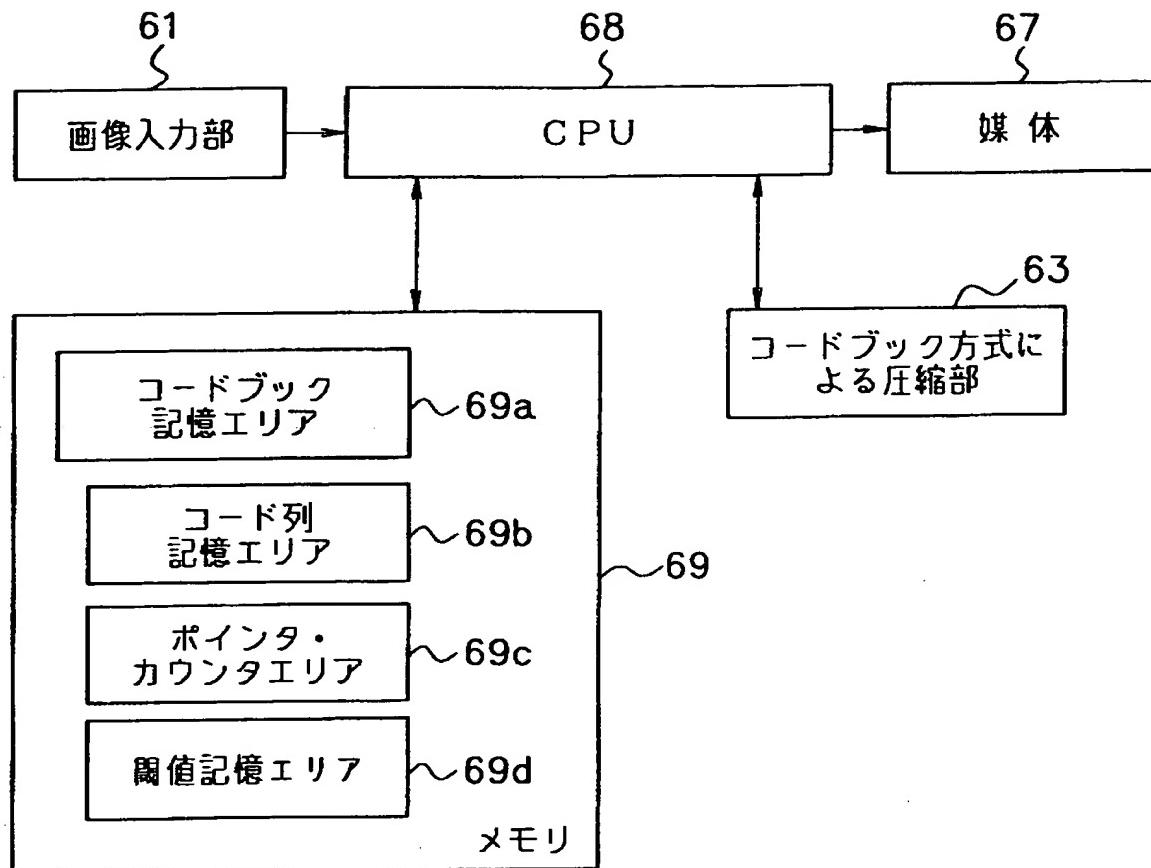
【図16】



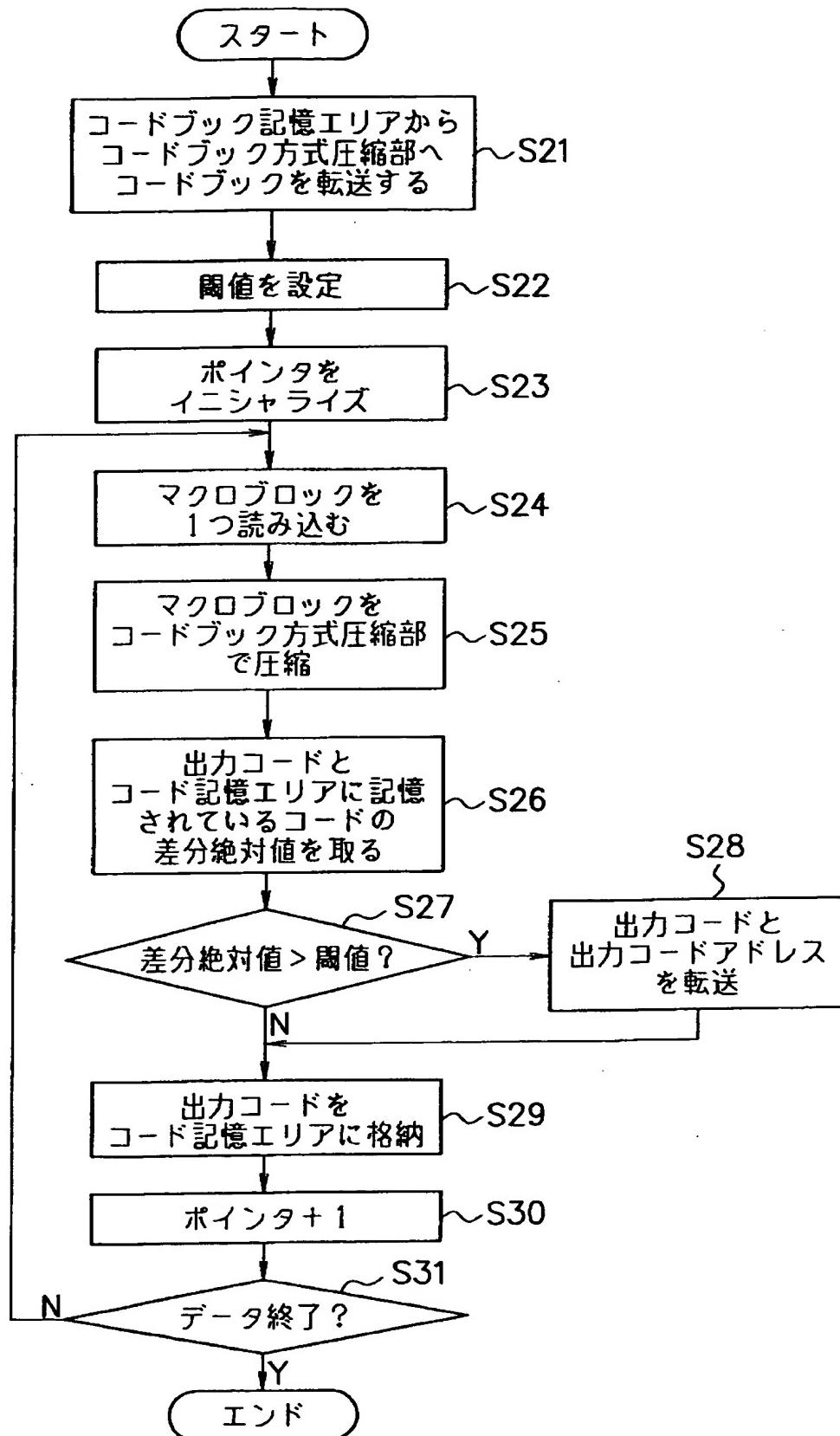
【図17】



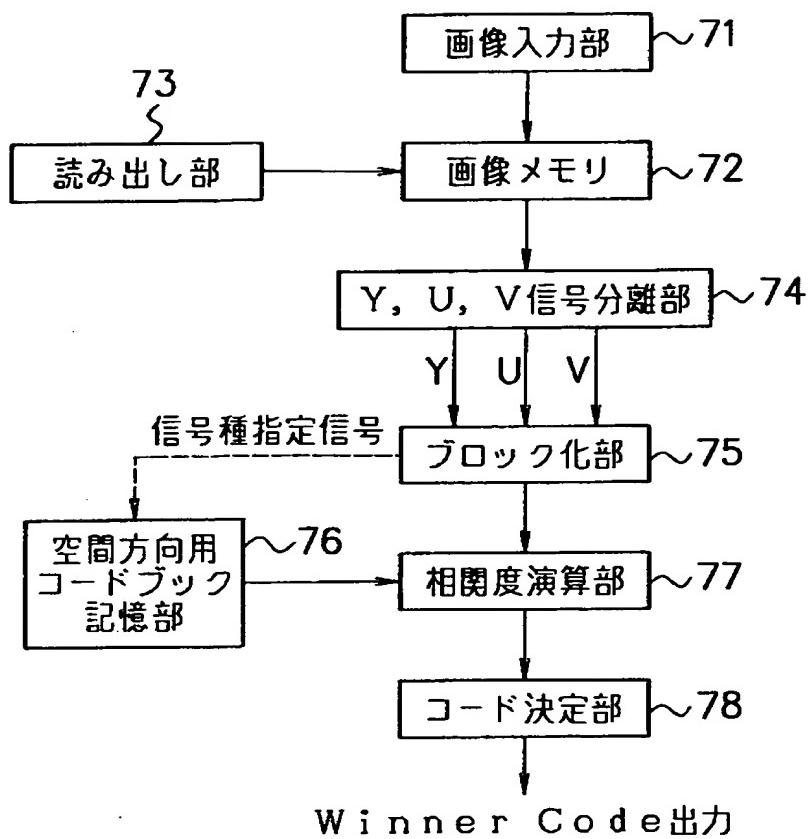
【図18】



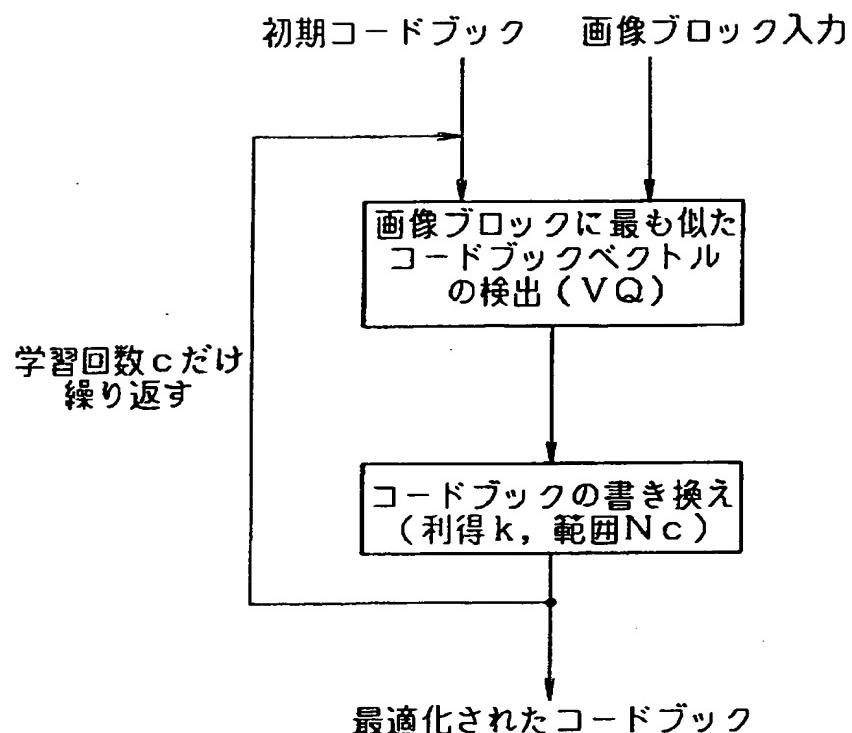
【図19】



【図20】



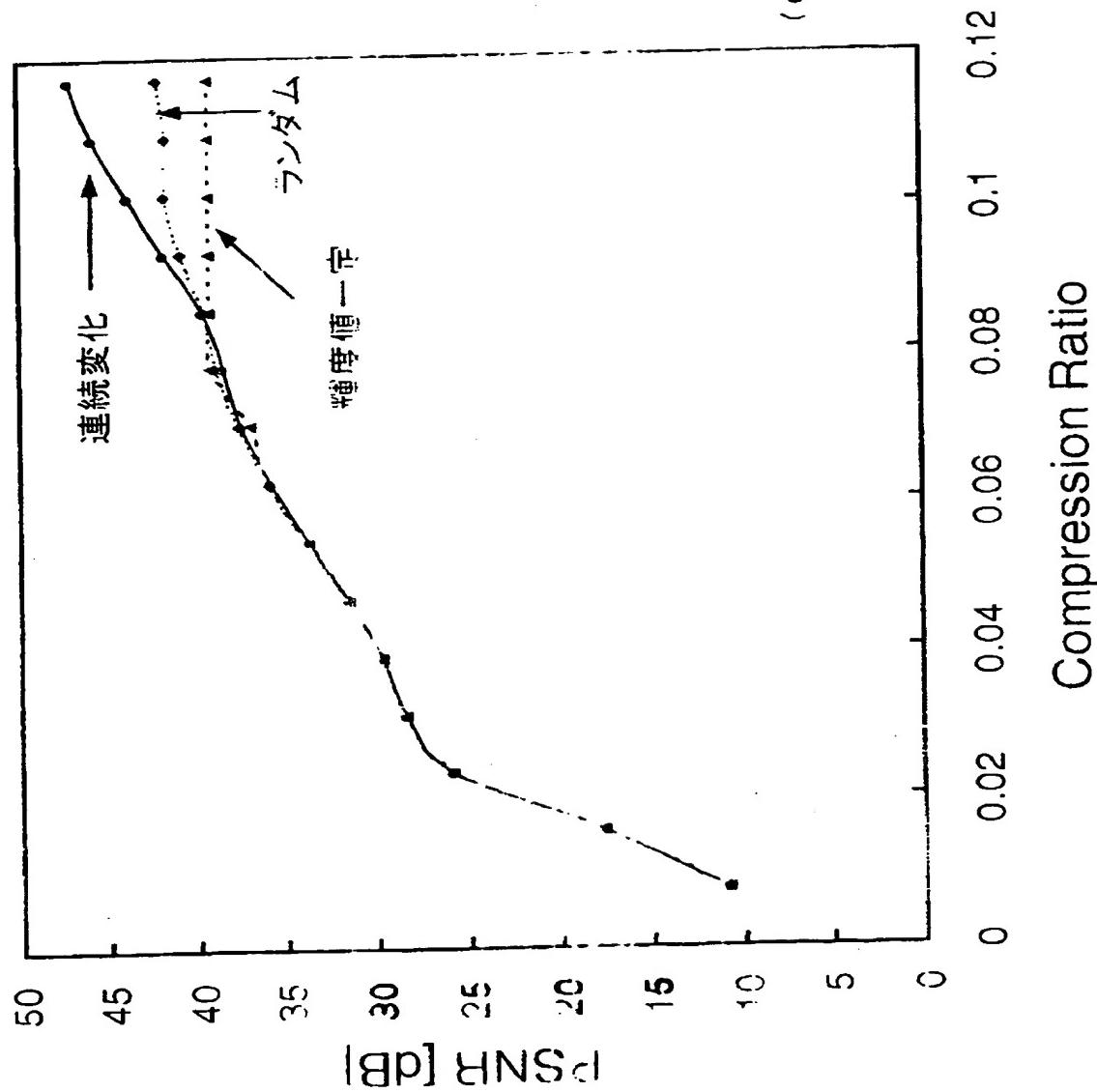
【図21】



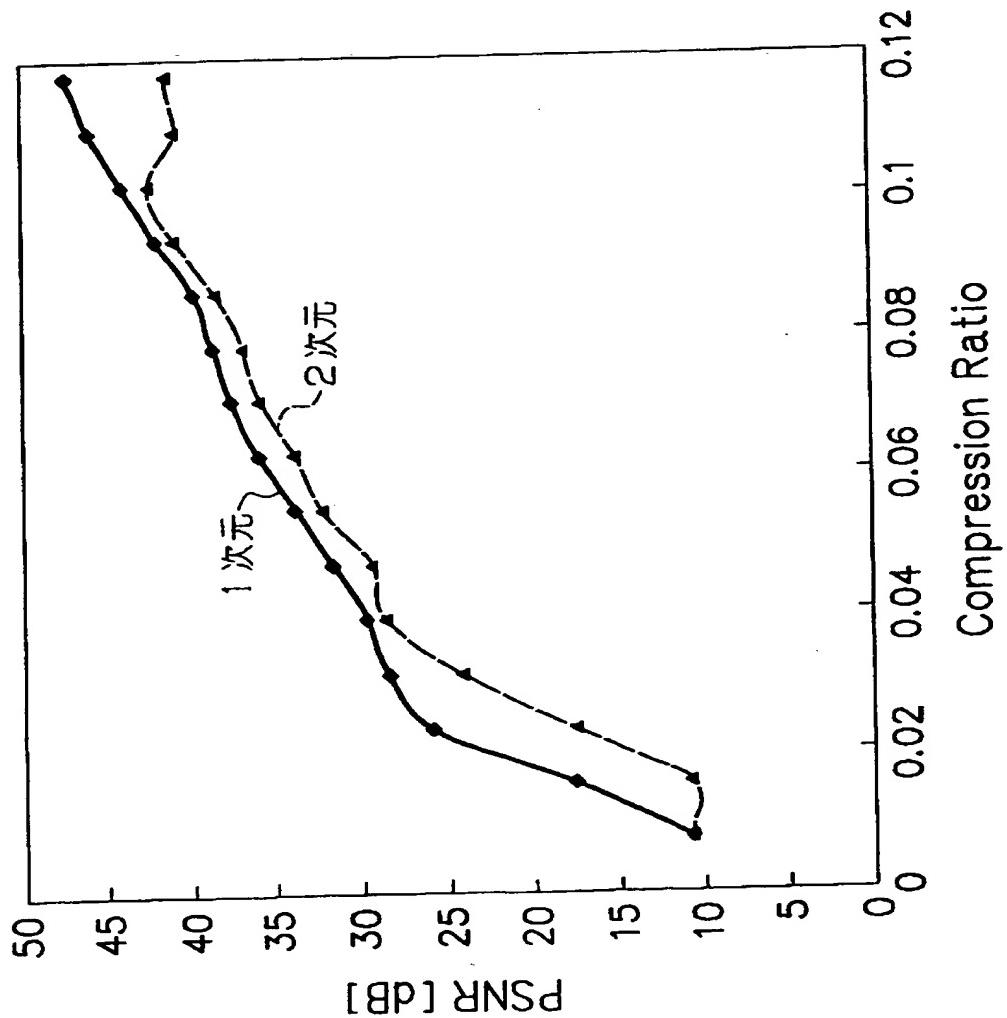
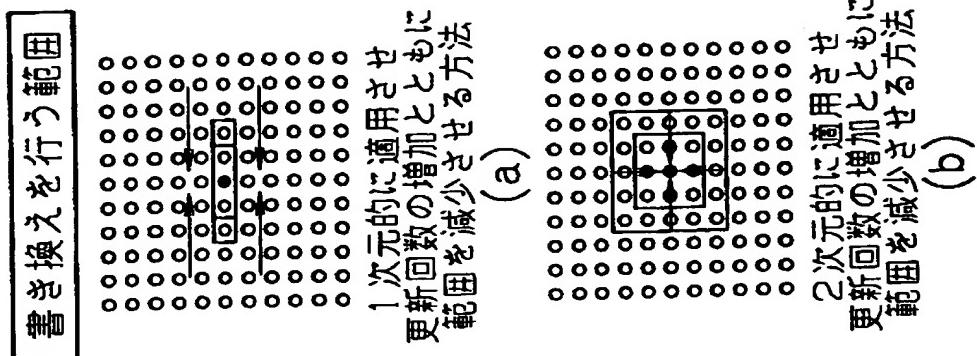
特平10-305336

【図22】

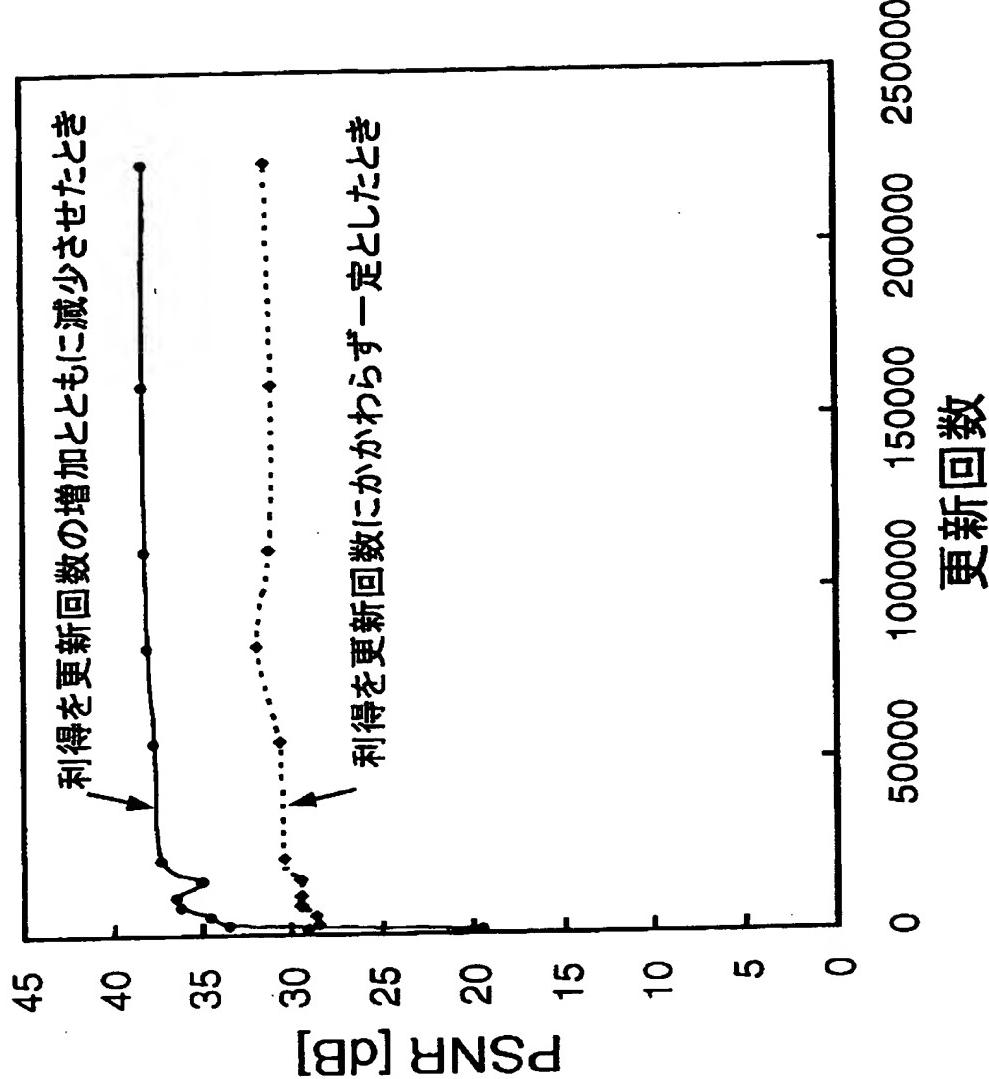
## 初期コードブックの種類



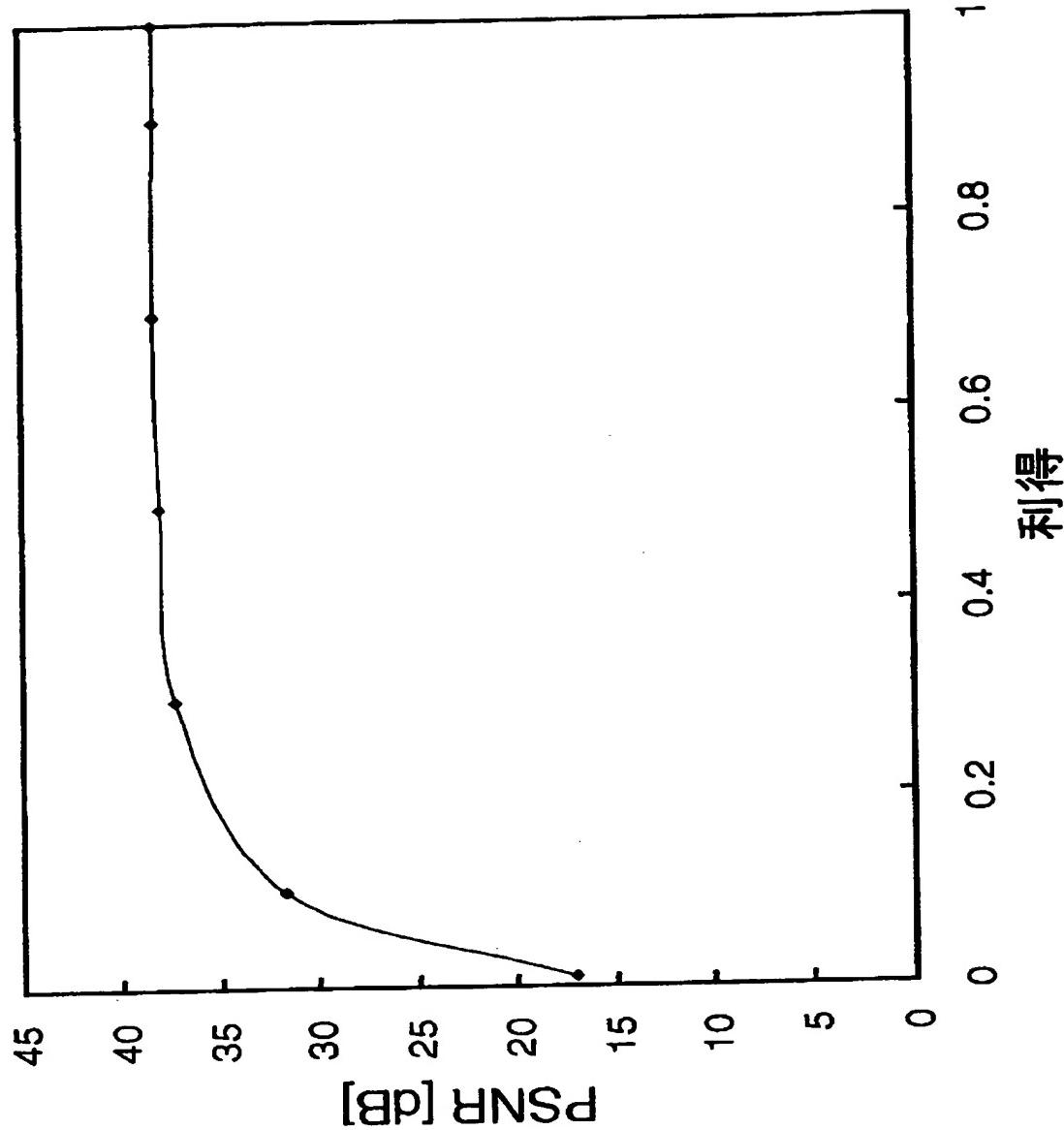
【図23】



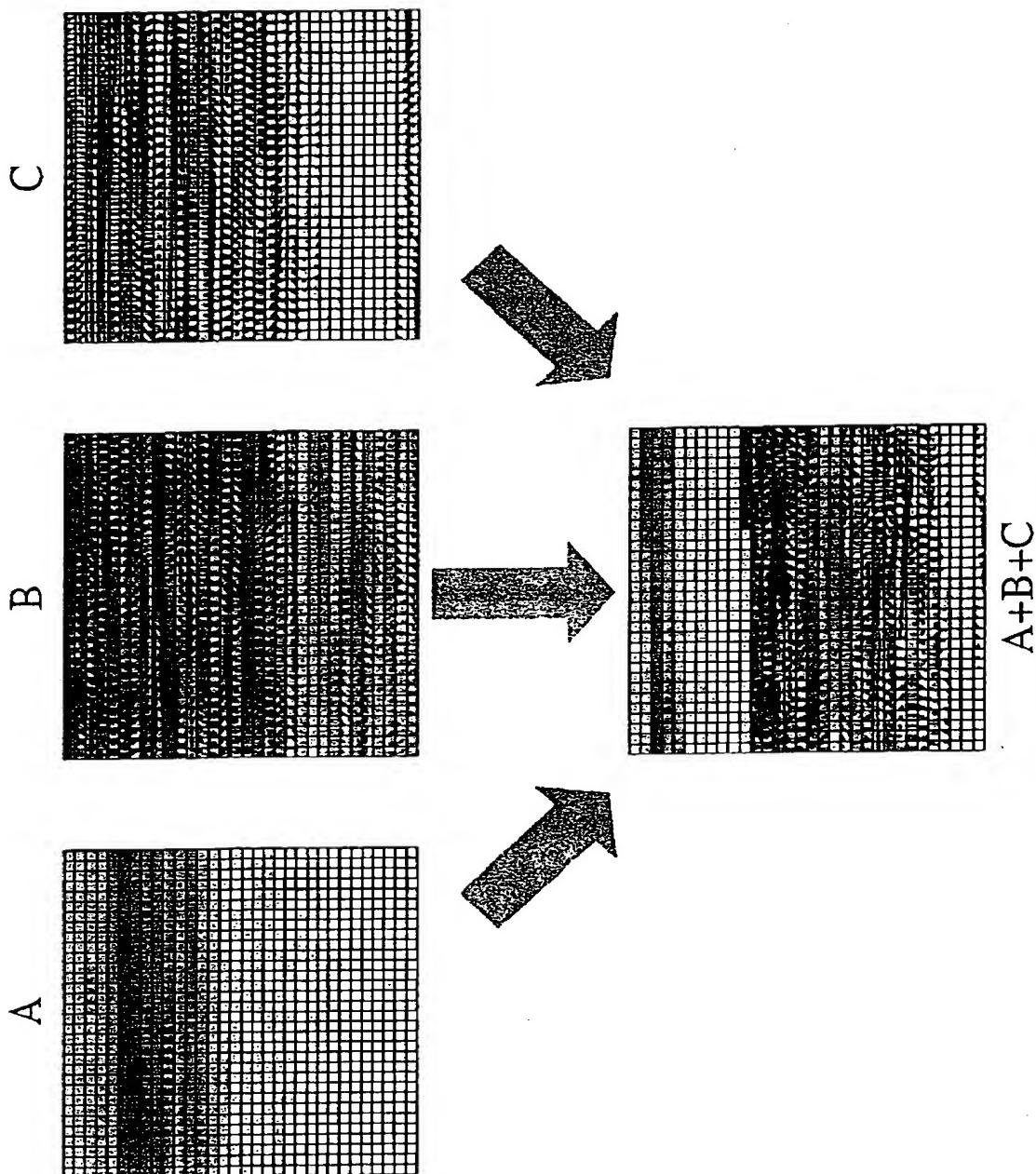
【図24】



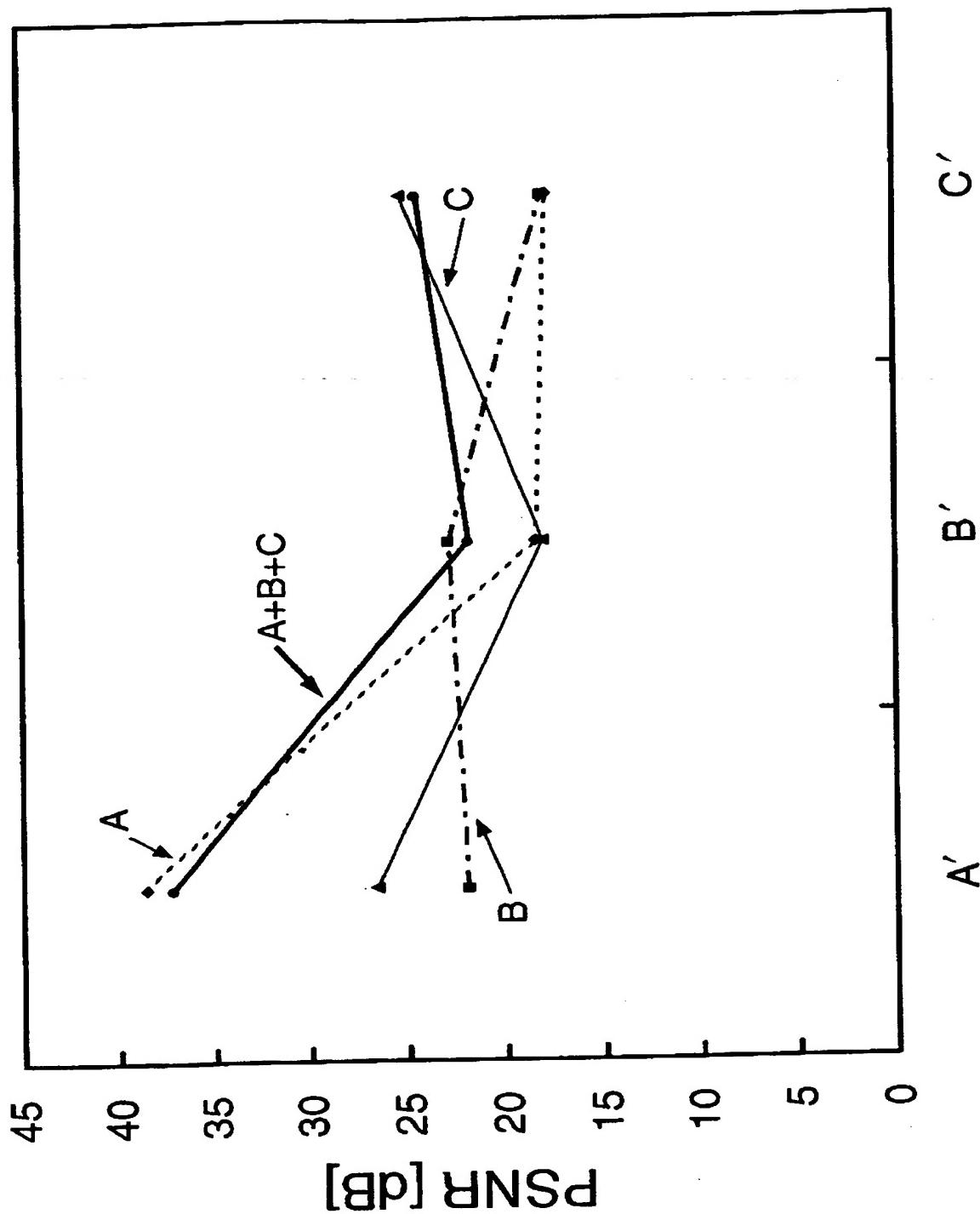
【図25】



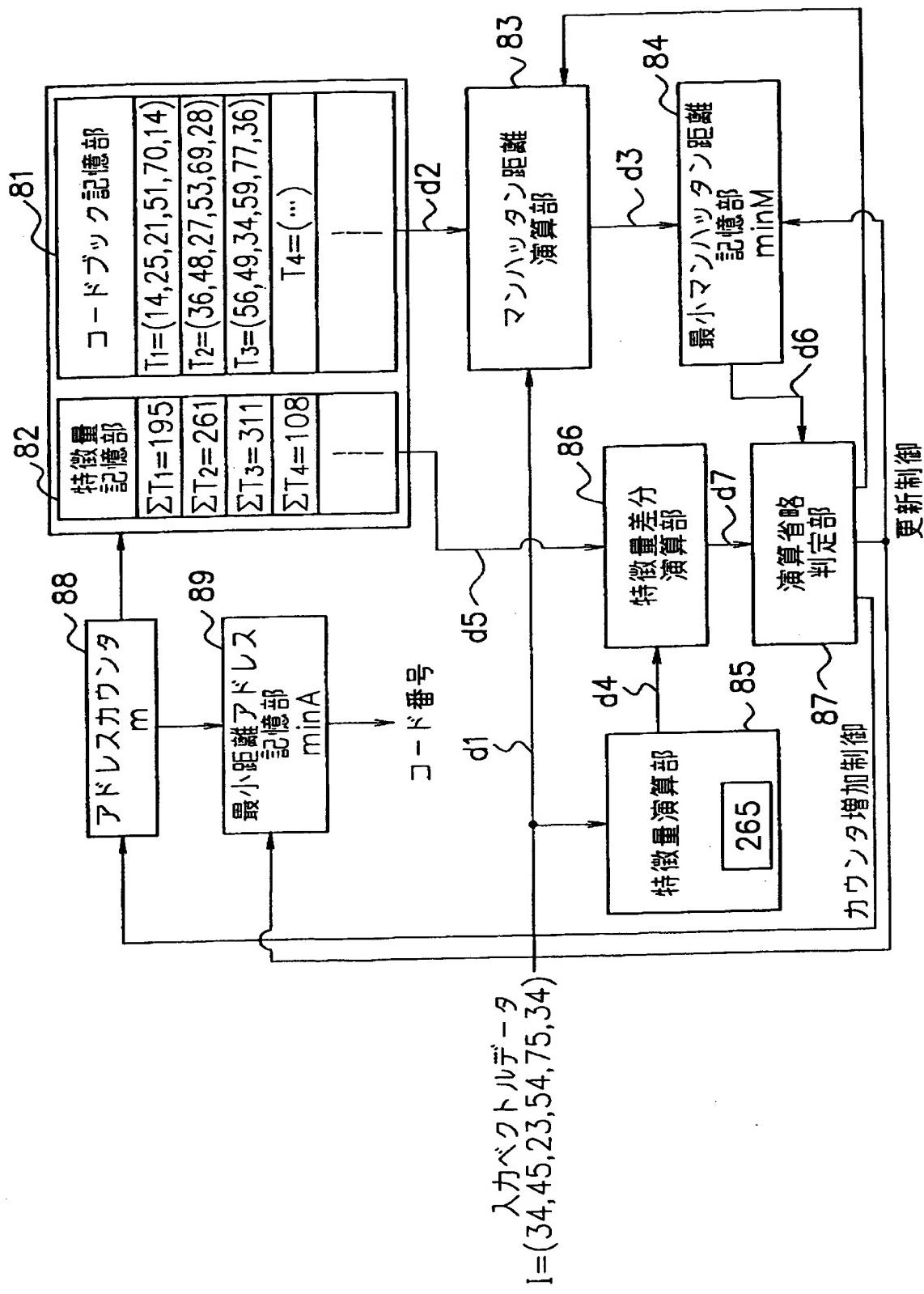
【図26】



【図27】

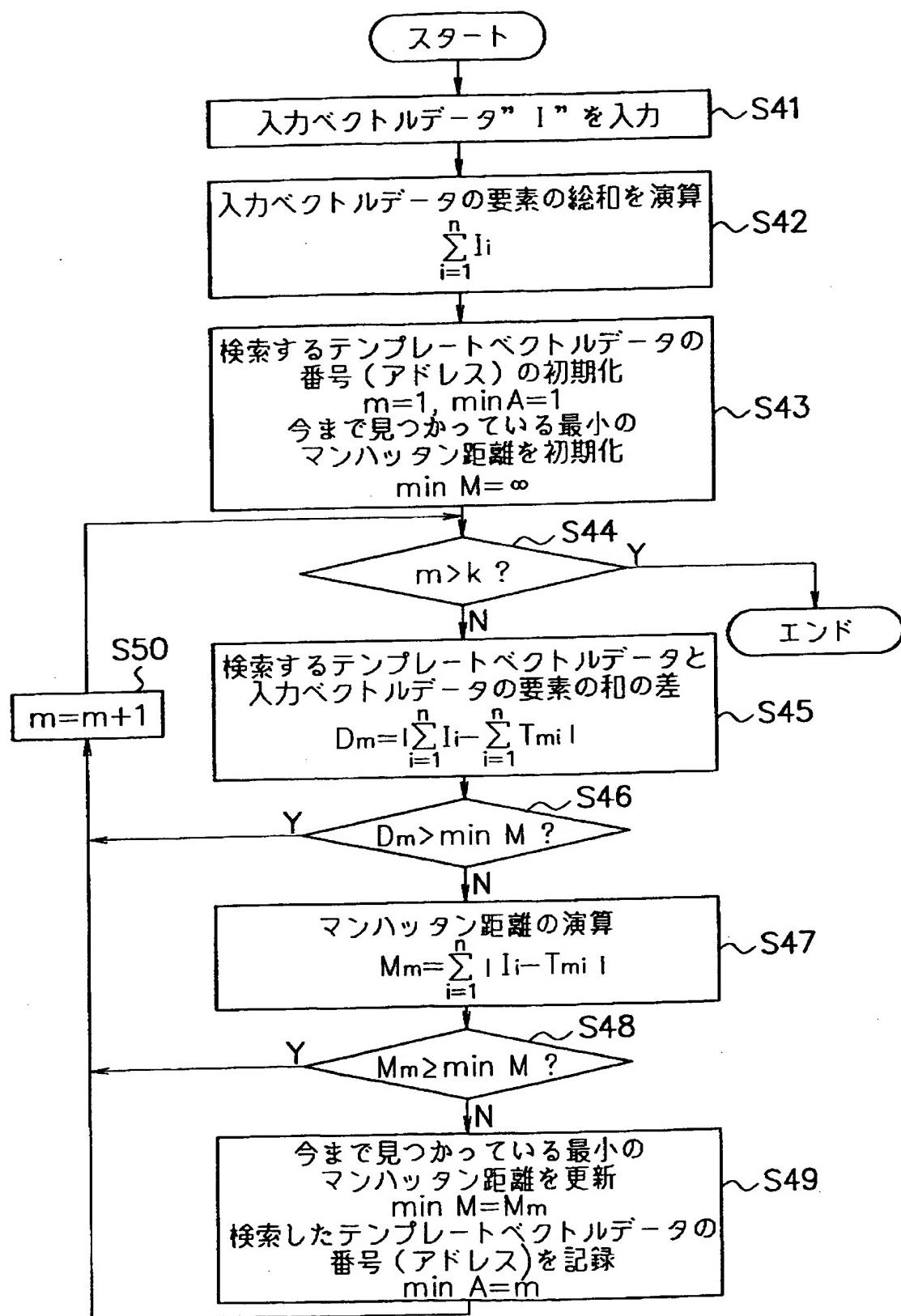


【図28】

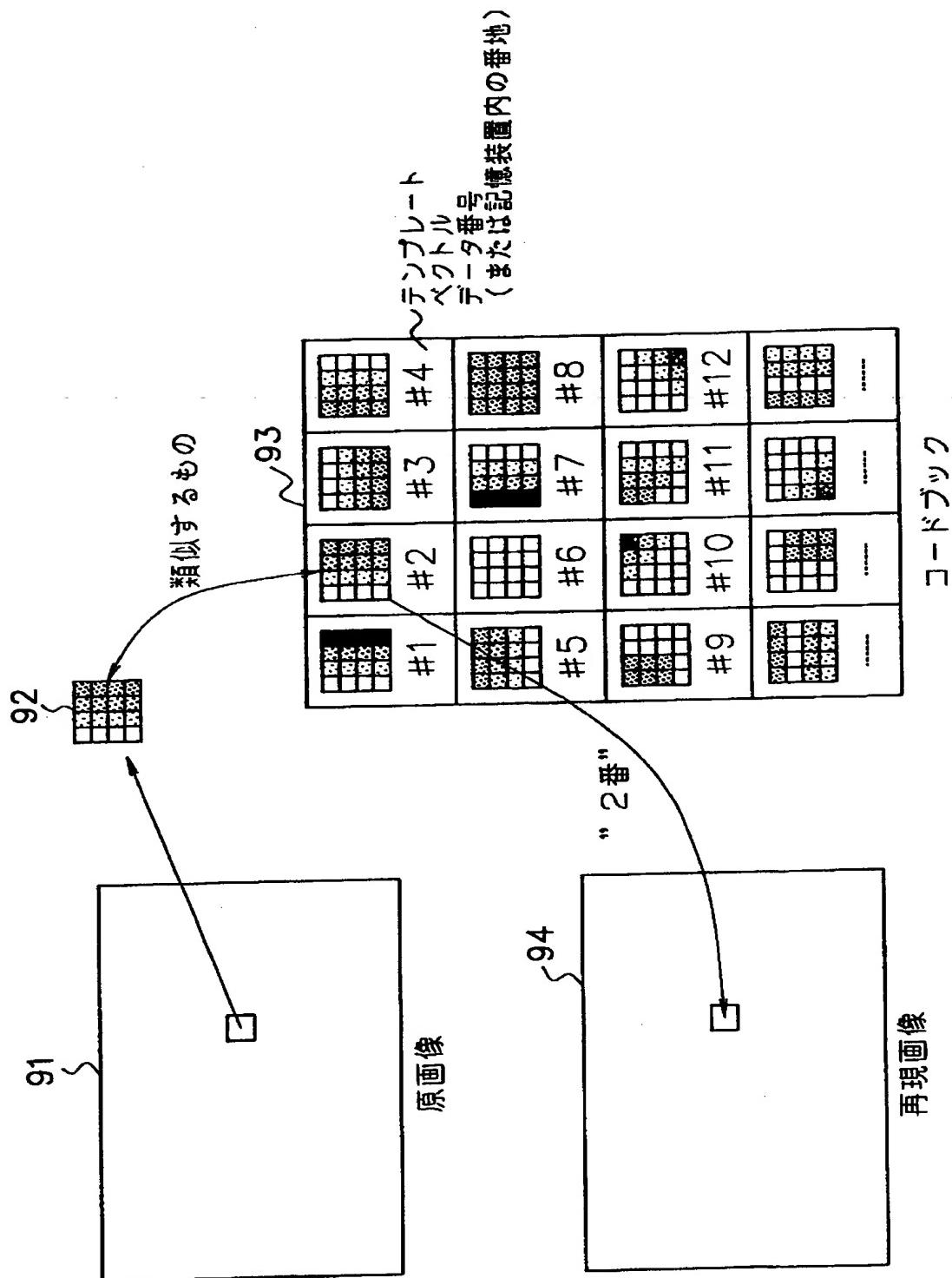


特平10-305336

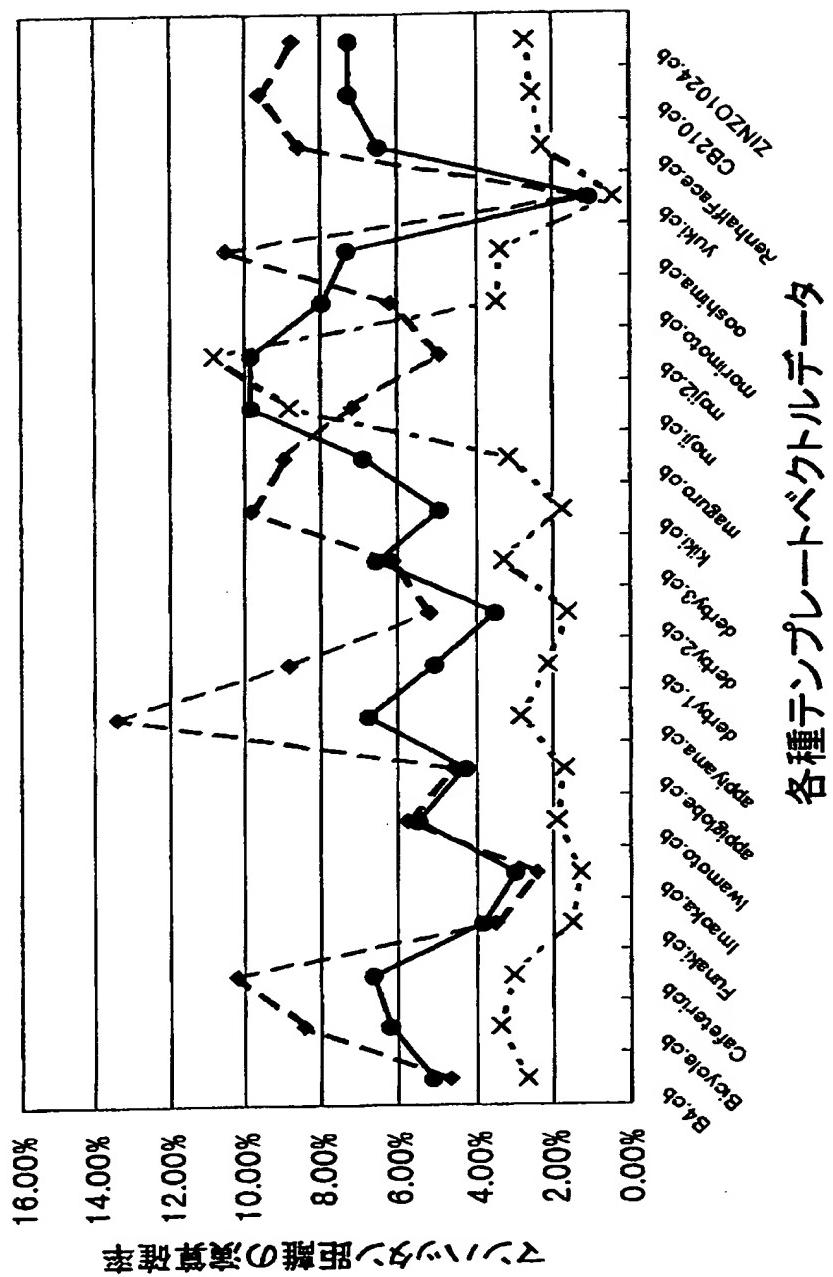
【図29】



【図30】



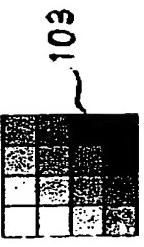
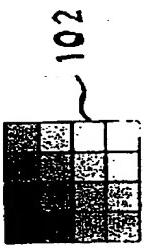
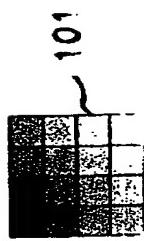
【図31】



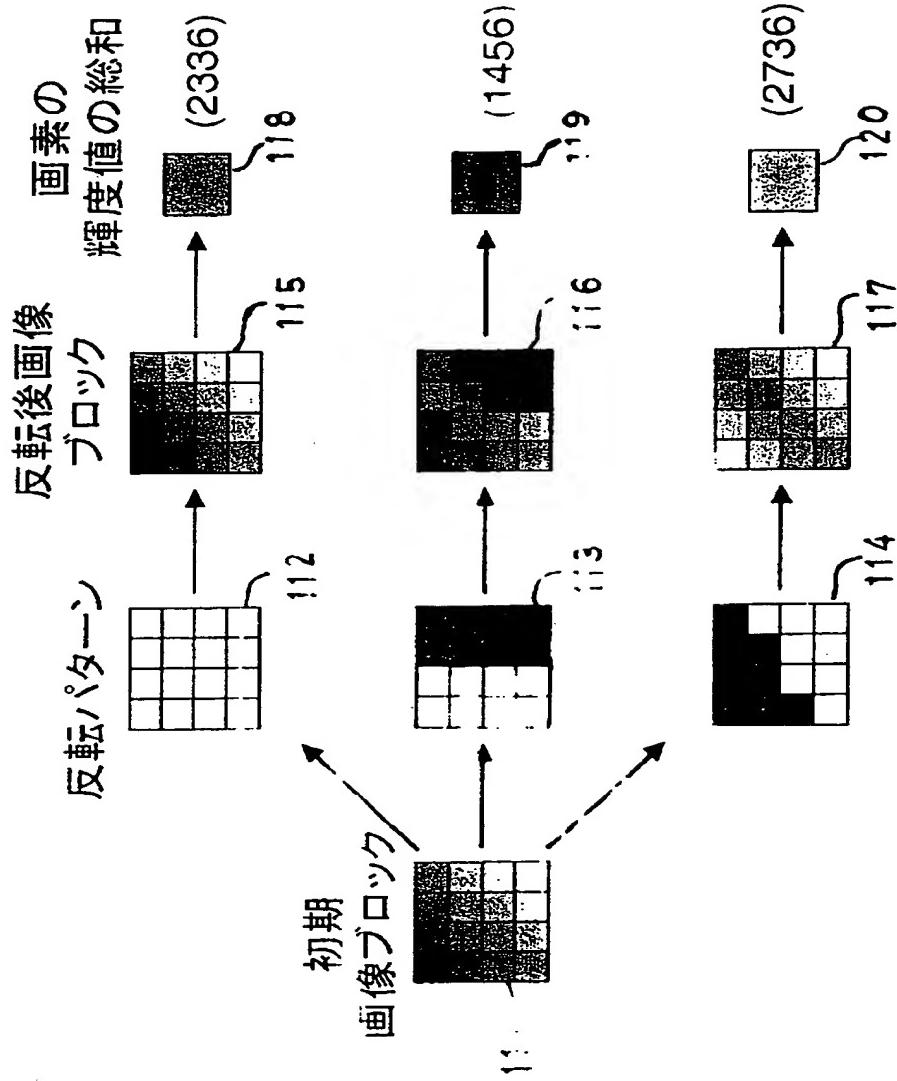
【図32】

入力ベクトルデータに  
対応する画像ブロック

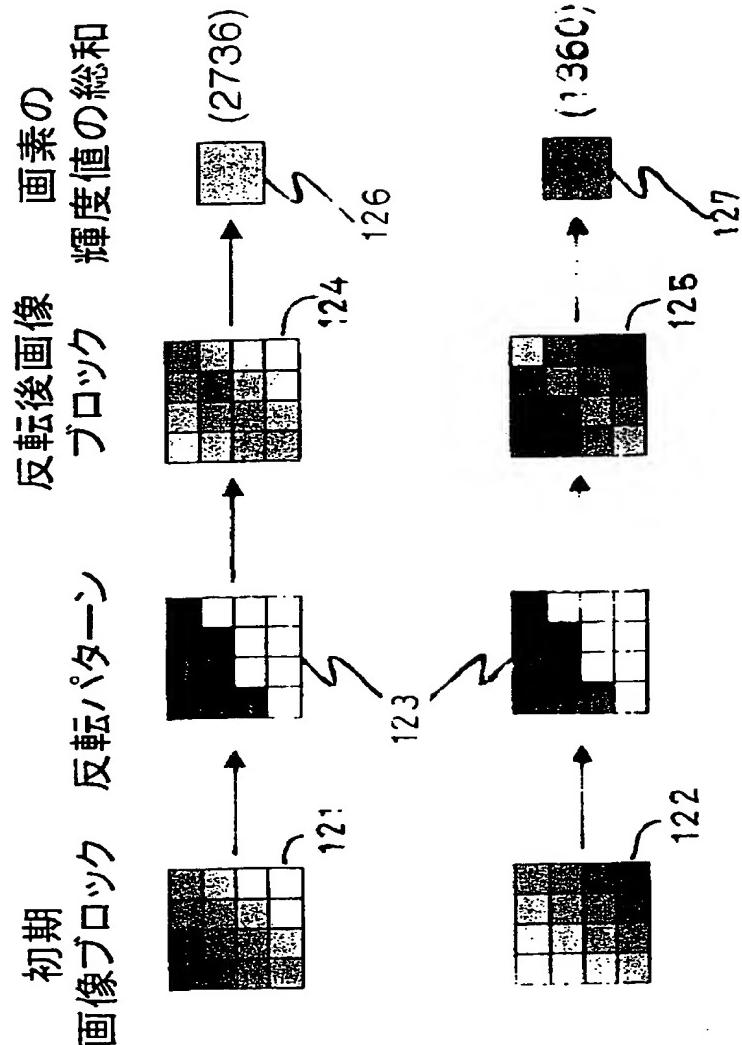
テンプレートベクトルデータに  
対応する画像ブロック



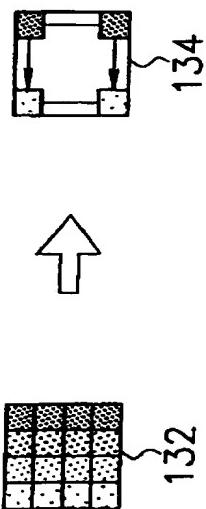
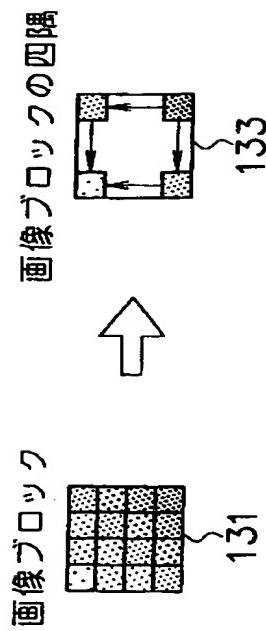
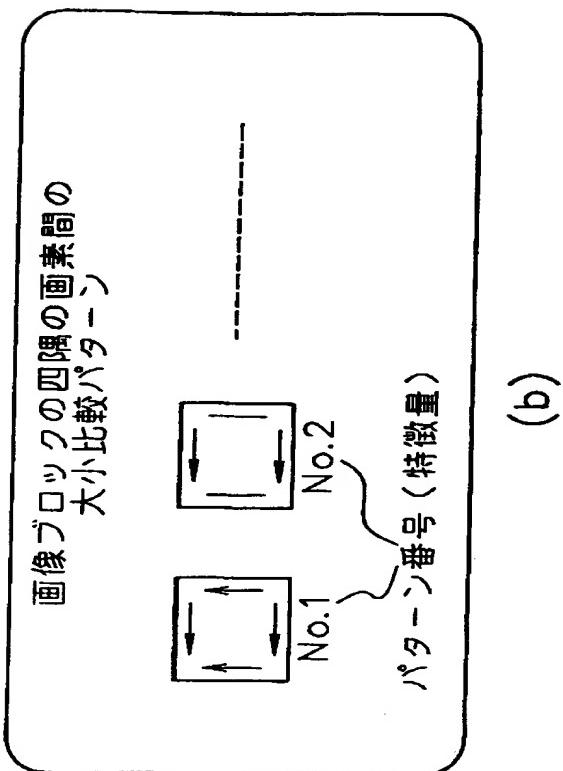
【図33】



〔図34〕



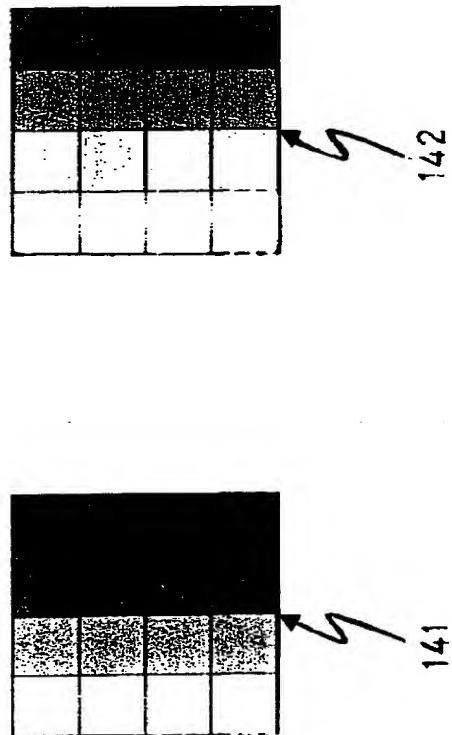
【図35】



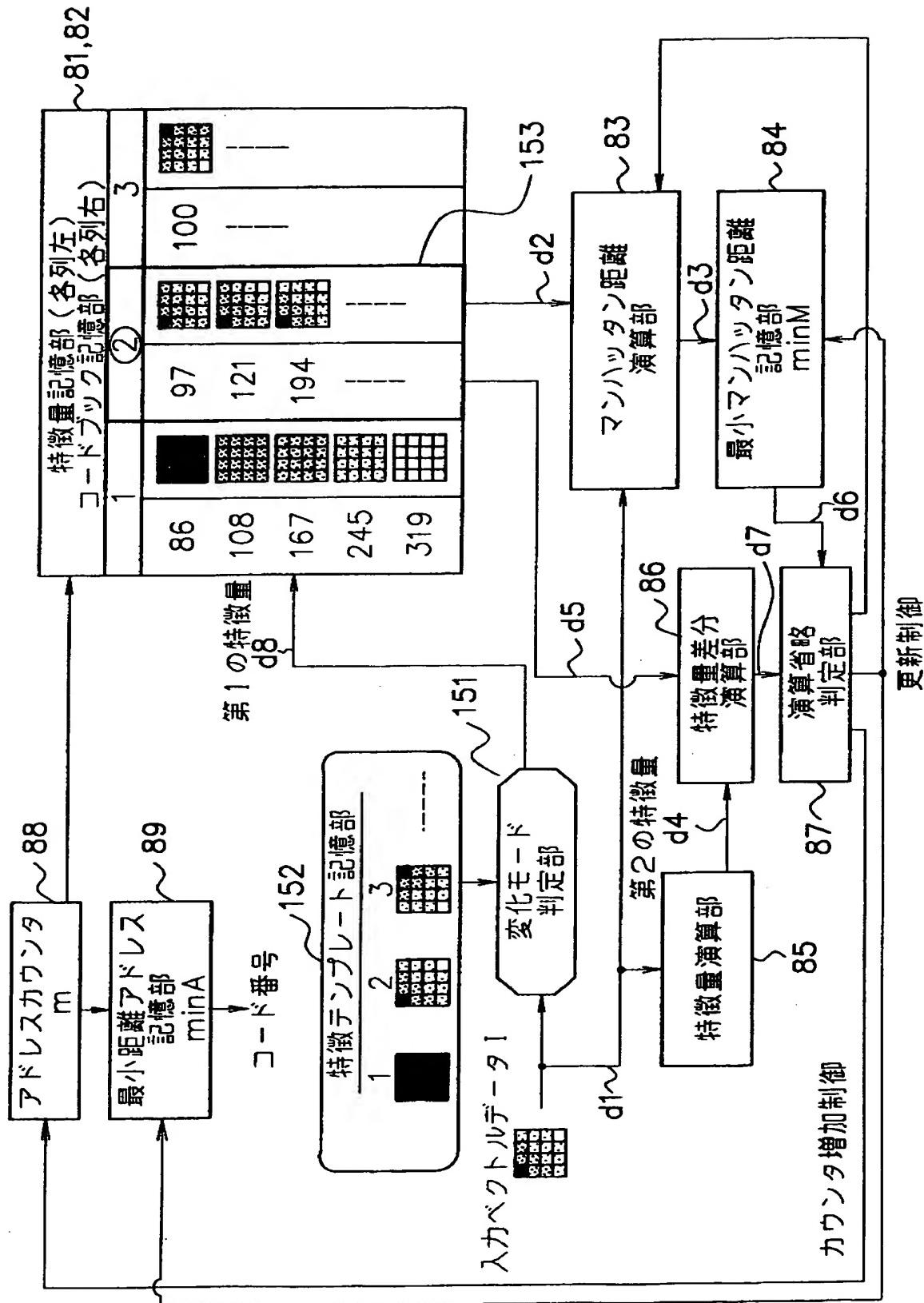
(a)

【図36】

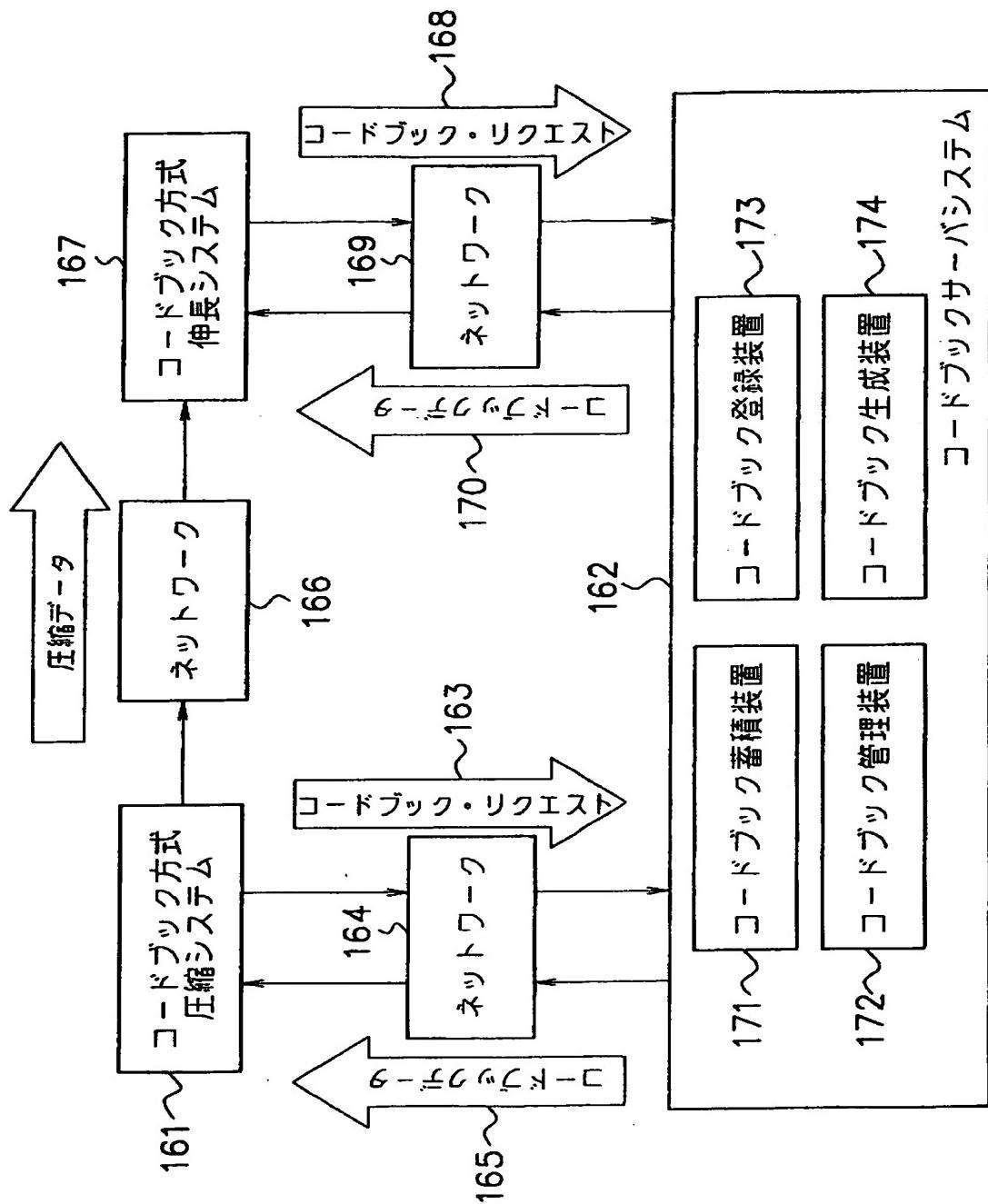
画像ブロック



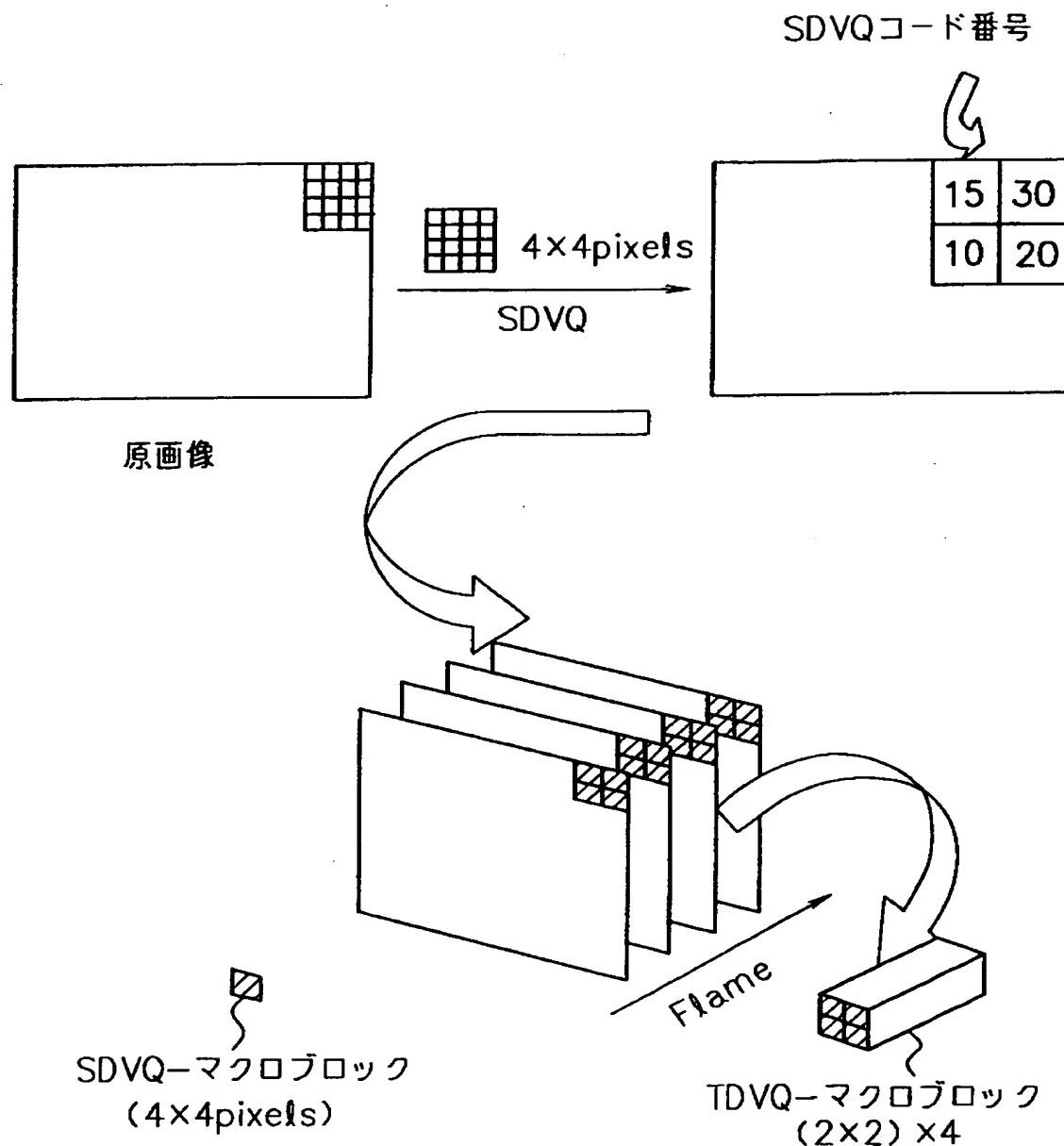
【図37】



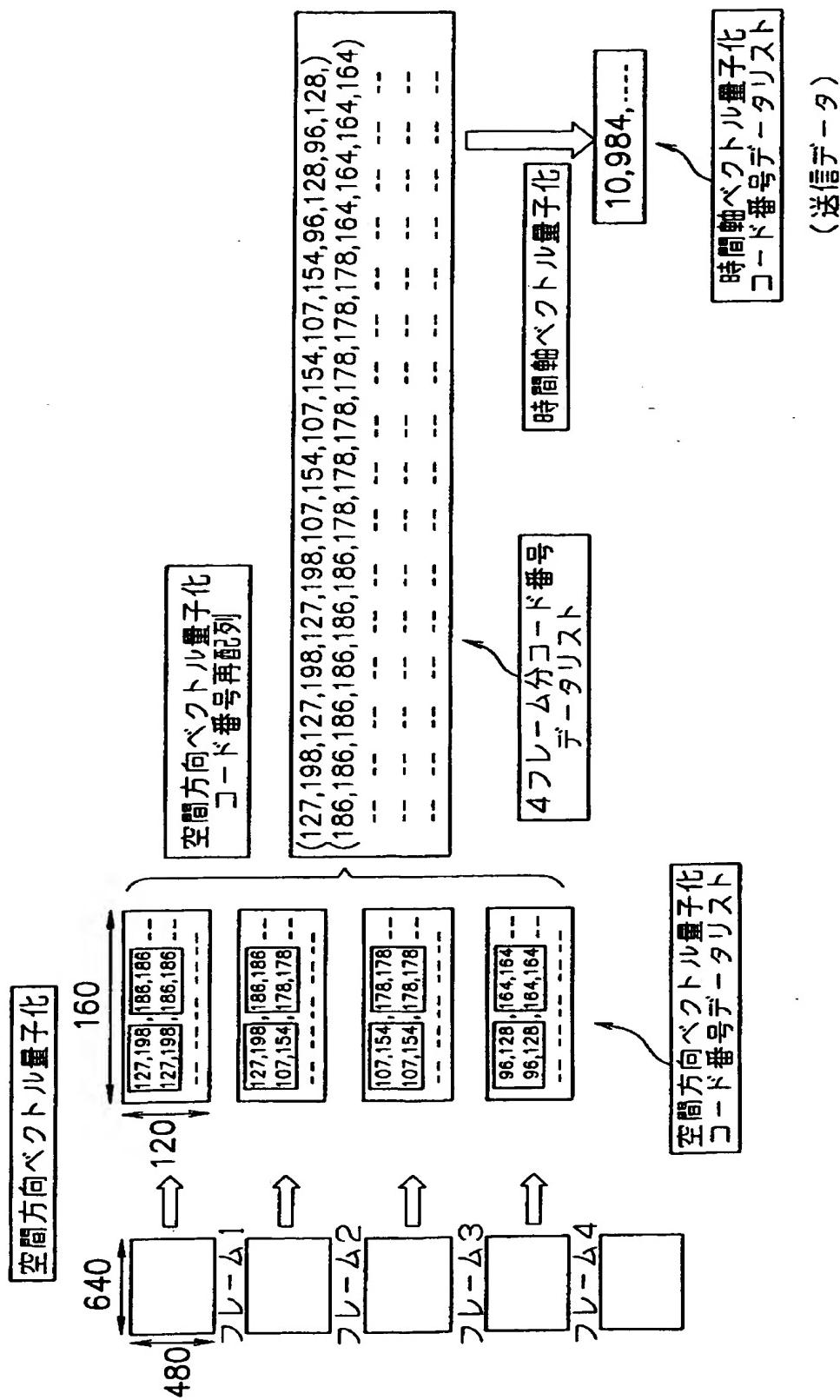
【図38】



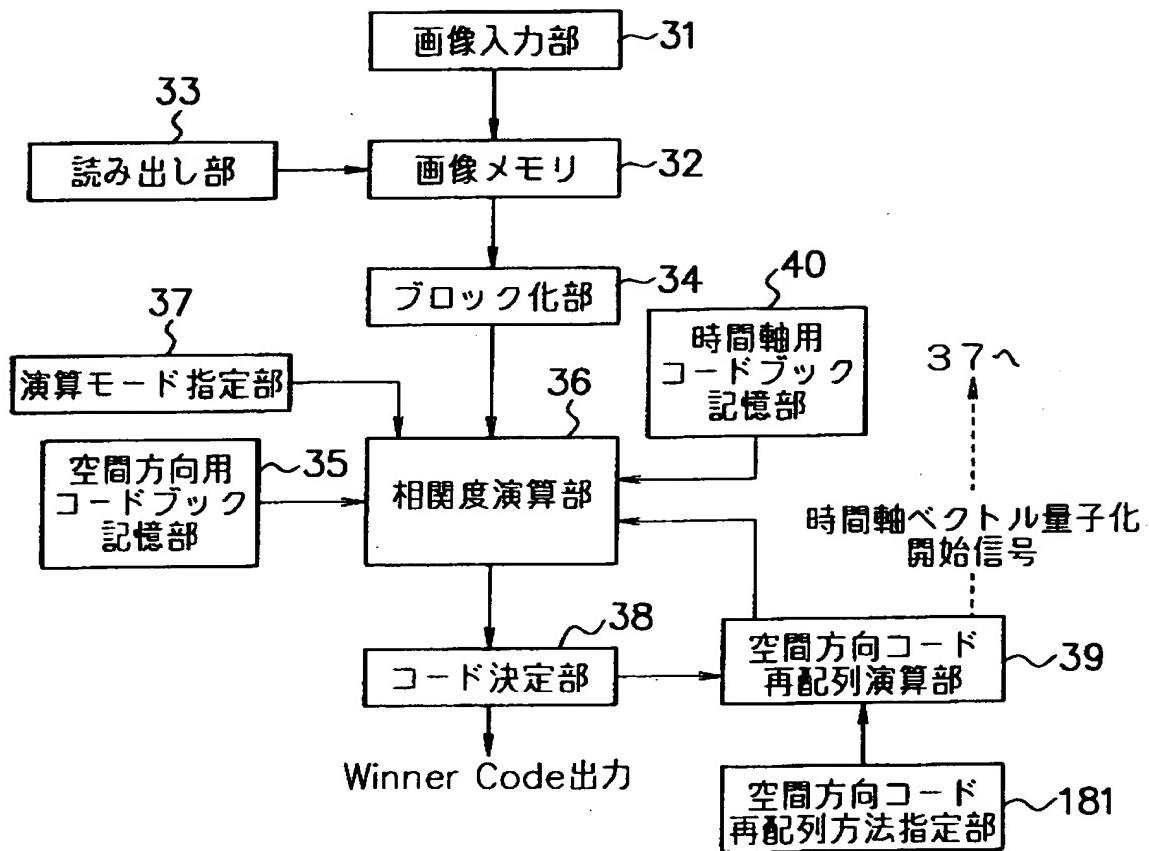
【図39】



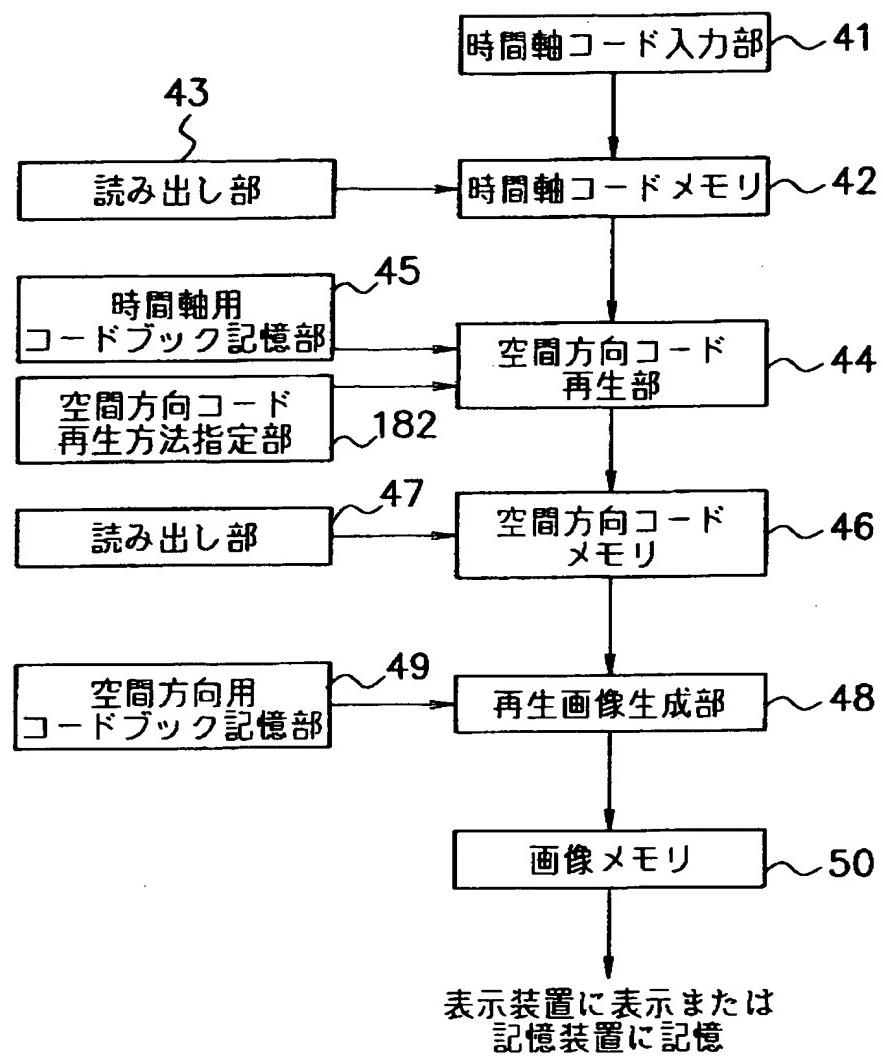
【図40】



【図41】

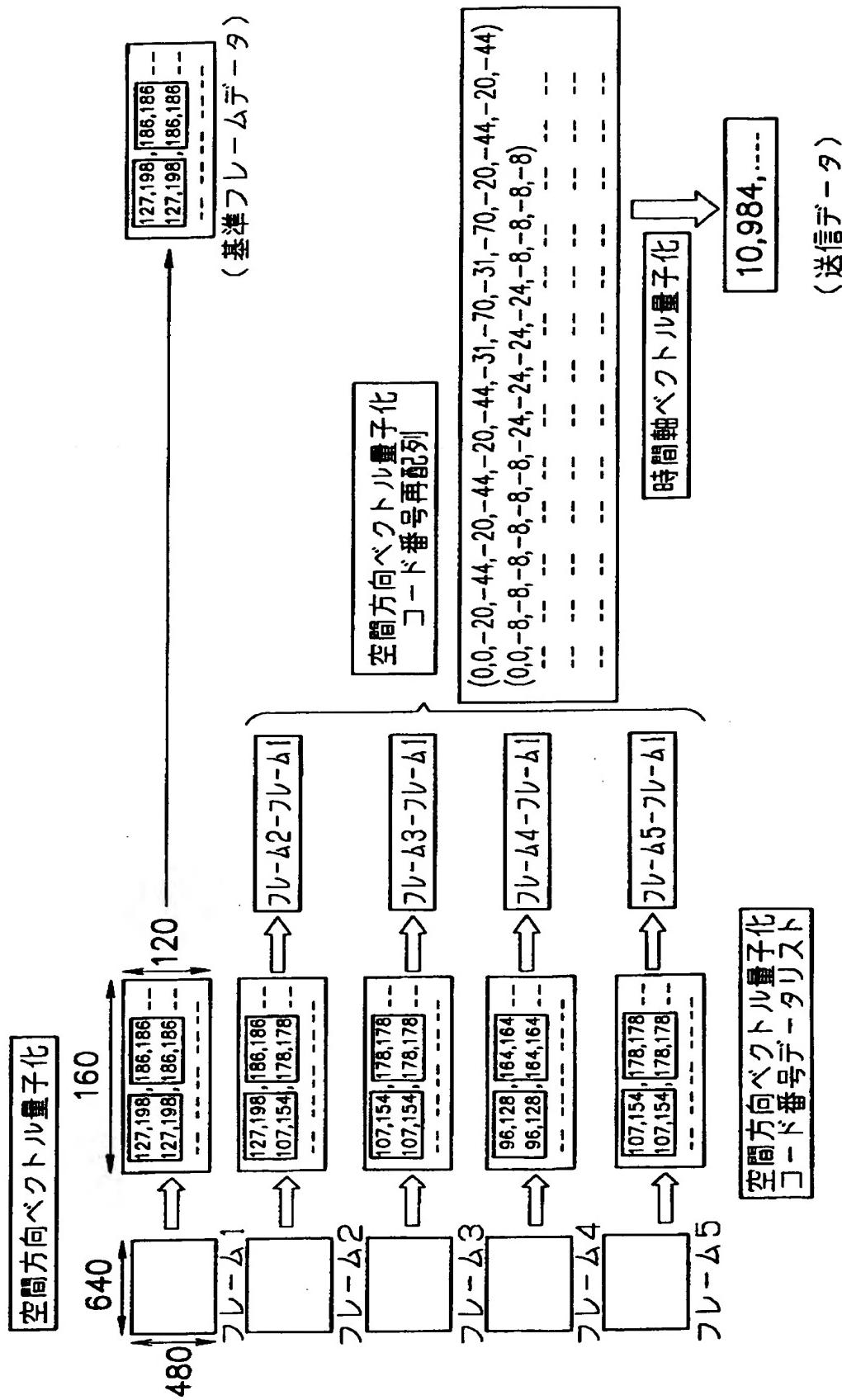


【図42】



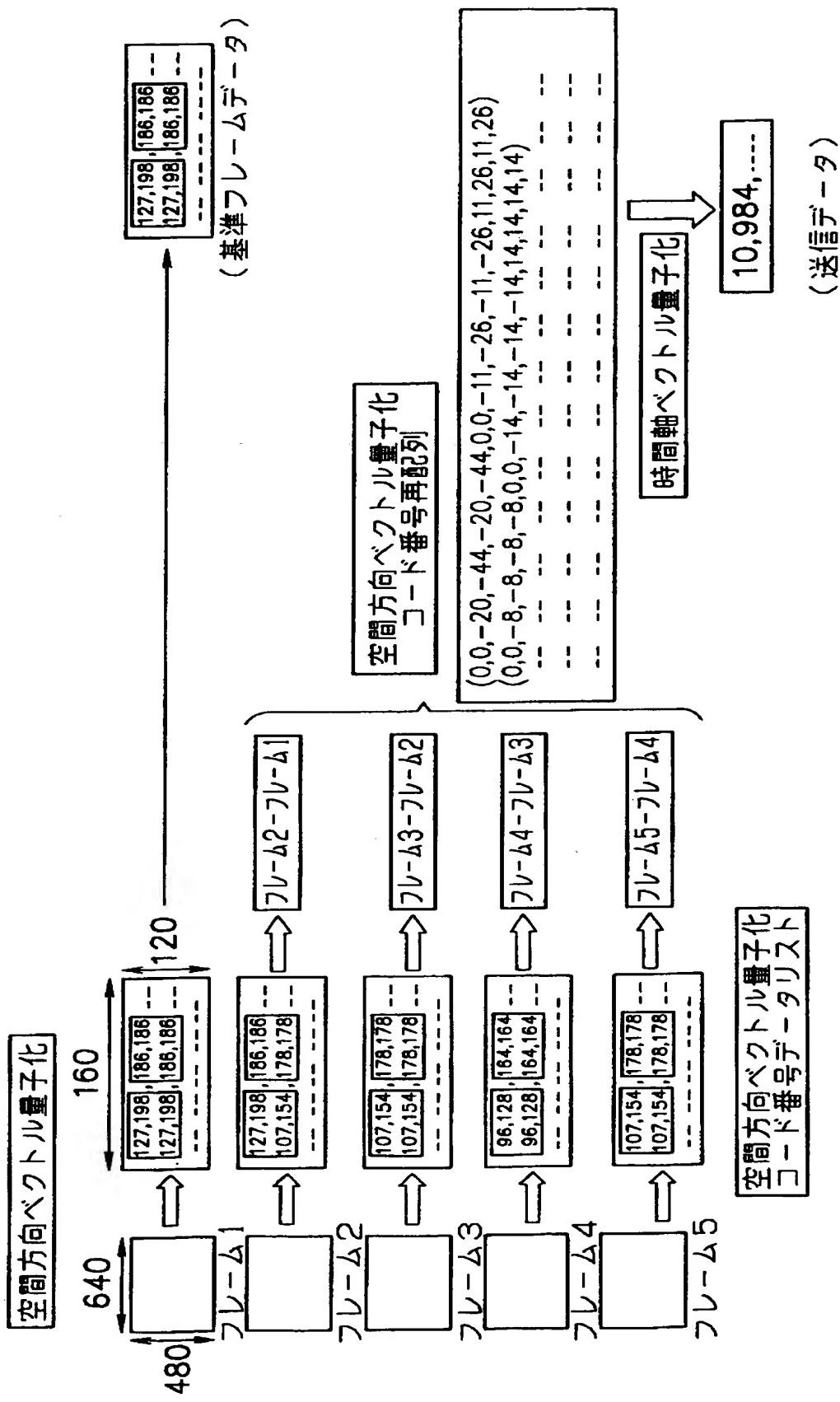
特平10-305336

【図43】

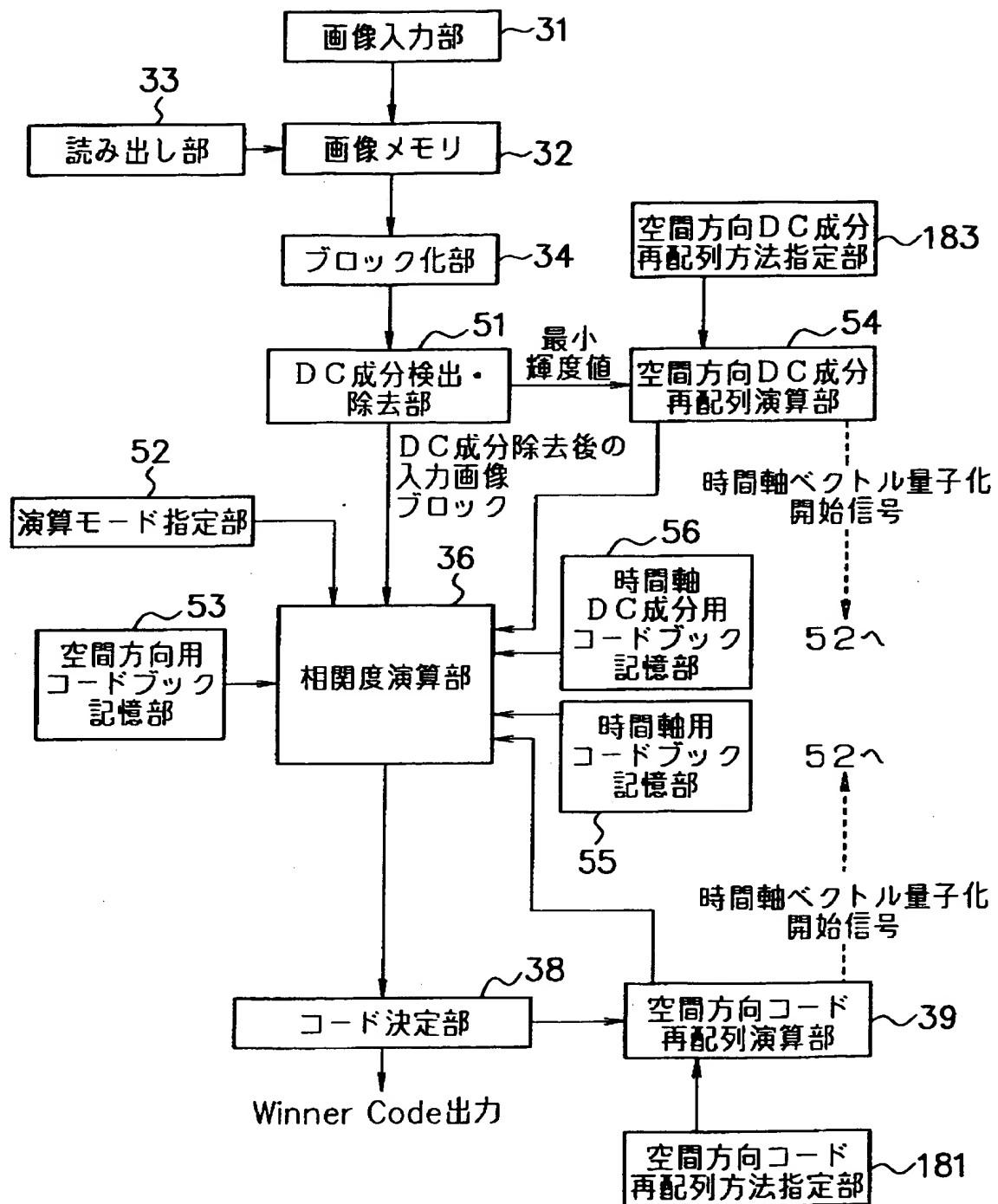


特平10-305336

【図44】



【図45】



【図46】

空間方向コード再配列の方法と圧縮率の関係

| 空間方向コードのマクロブロックの取り方 | 圧縮対象フレーム | 圧縮率      |
|---------------------|----------|----------|
| 1×1                 | 16フレーム   | 341分の1   |
| 2×2                 | 4フレーム    | 341分の1   |
| 4×1                 | 4フレーム    | 341分の1   |
| 4×4                 | 2フレーム    | 341分の1   |
| 3×3                 | 3フレーム    | 576分の1   |
| 3×3                 | 4フレーム    | 768分の1   |
| 4×4                 | 4フレーム    | 1365分の1  |
| 6×6                 | 6フレーム    | 4608分の1  |
| 8×8                 | 8フレーム    | 10922分の1 |

【書類名】 要約書

【要約】

【課題】 画像や音声のデータ圧縮を高圧縮率で実現し、圧縮データをもとに品質の高いデータを再生できるようにする。

【解決手段】 相関度演算部36を、動画の各フレーム画像に対して空間方向用コードブックを用いて個々にベクトル量子化するとともに、得られたコード列を空間方向コード再配列演算部39で再配列して複数の新たなベクトルとし、それらの新たなベクトルに対して更に時間軸用コードブックを用いてベクトル量子化するように構成し、1つのフレーム画像に対して空間方向に対するベクトル量子化を行うだけでなく、時間軸に沿った複数のフレーム間で時間軸方向に対するベクトル量子化も行うようにすることにより、各フレーム間に対してもデータ圧縮を行うことができるようにして、再生画像の品質を維持したままより高い圧縮率で動画を圧縮することができるようとする。

【選択図】 図9

【書類名】 職権訂正データ  
【訂正書類】 特許願

<認定情報・付加情報>

【特許出願人】

【識別番号】 000205041

【住所又は居所】 宮城県仙台市青葉区米ヶ袋2-1-17-301

【氏名又は名称】 大見 忠弘

【特許出願人】

【識別番号】 596089517

【住所又は居所】 東京都文京区本郷4-1-4

【氏名又は名称】 株式会社ウルトラクリーンテクノロジー開発研究所

【代理人】 申請人

【識別番号】 100090273

【住所又は居所】 東京都豊島区東池袋1丁目17番8号 池袋TGホ  
ームズビル5階 國分特許事務所

【氏名又は名称】 國分 孝悦

出願人履歴情報

識別番号 [000205041]

1. 変更年月日 1990年 8月27日

[変更理由] 新規登録

住 所 宮城県仙台市青葉区米ヶ袋2-1-17-301  
氏 名 大見 忠弘

出願人履歴情報

識別番号 [596089517]

1. 変更年月日 1996年 6月20日

[変更理由] 新規登録

住 所 東京都文京区本郷4-1-4

氏 名 株式会社ウルトラクリーンテクノロジー開発研究所